Fast Methods for Static Hamilton-Jacobi Partial Differential Equations

by

Alexander Boris Vladimirsky

B.A. degree (University of California, Berkeley) 1995

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Applied Mathematics

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor James A. Sethian, Chair
Professor Alexandre J. Chorin
Professor Alistair Sinclair

Spring 2001

Fast Methods for Static Hamilton-Jacobi Partial Differential Equations

Abstract

Fast Methods for Static Hamilton-Jacobi Partial Differential Equations

by

Alexander Boris Vladimirsky

Doctor of Philosophy in Applied Mathematics

University of California at BERKELEY

Professor James A. Sethian, Chair

We develop a family of fast methods approximating the solution to a wide class of static Hamilton-Jacobi partial differential equations. These partial differential equations are considered in the context of control-theoretic and front-propagation problems.

In general, to produce a numerical solution to such a problem, one has to solve a large system of coupled non-linear discretized equations. Our techniques use partial information about the characteristic directions to de-couple the system.

Previously known fast methods, available for isotropic problems, are discussed in detail. We introduce a family of new Ordered Upwinding Methods (OUM) for general (anisotropic) problems and prove convergence to the viscosity solution of the corresponding Hamilton-Jacobi partial differential equation. The hybrid methods introduced here are based on our analysis of the role played by anisotropy in the context of front propagation and optimal trajectory problems.

The performance of the methods is analyzed and compared to that of several other numerical approaches to these problems. Computational experiments are performed using test problems from control theory, computational geometry and seismology.

I was very lucky to meet many talented and enthusiastic mathematicians, who have greatly influenced the scope and the overall direction of my mathematical interests. However, long before I got to my first University, there were several people, whose views had determined my basic attitude towards this subject.
I would like to dedicate this work to three of them.

To my father and friend, Boris Vladimirsky,
who taught me that knowing something is no substitute for understanding it.

To my teacher and friend, Larisa Polushkina,
who taught me to appreciate the beauty of mathematics.

To my friend and mentor, Alexander Dukhovny,
who taught me to look for the paths of my own
in choosing which problems to solve and in trying to solve them.

Contents

List of Figures

Acknowledgements

First things first.
I am very grateful to the members of my dissertation committee, James Sethian, Alexandre Chorin, and Alistair Sinclair, for agreeing to read through this treatise. Writing it was an experience both frustrating and exhilarating.

In 1993 I came to UC Berkeley to complete the last two years of my undergraduate degree. At that time my plans for the future were rather vague and ranged from founding a multinational corporation to pursuing a graduate degree in algebraic geometry. I am glad to have this occasion to thank John Strain and William Kahan, encounters with whom have convinced me that I want to become an Applied Mathematician and that Berkeley is **the** place to do that.

Upon entering the Ph.D. program in 1995, I have taken James Sethian's seminar course on numerical methods for partial differential equations. The general discourse of that seminar had largely determined my research interests for the last six years. The majority of the results presented in this thesis were obtained in the pursuit of extending the ideas behind Sethian's Fast Marching Method to a wider class of PDEs and application domains. Several other participants in that course have also greatly contributed to my perspective on these methodologies. I would like to thank David Adalsteinsson, Ron Kimmel and Ravi Malladi for their enthusiastic and detailed responses to my inquiries about various flavors, extensions, and applications of the Fast Marching Method.

For the five out of six years spent in the graduate program, I have also enjoyed a friendly atmosphere and productive work environment of the Mathematics Department at Lawrence Berkeley National Laboratory. Being a part of this group was a uniquely valuable experience. It exposed me to a variety of perspectives on role of Mathematics in natural sciences, and convinced me of the ultimate interconnectedness of those perspectives.

My thanks to G.I. Barenblatt for explaining many topics both in classical mechanics and in modern politics. I am also grateful to him (and to Alexandre Chorin) for sharing a number of wonderful stories about the history of mathematics. It was very exciting to feel connected to the greater mathematical tradition.

I would like to thank all of the members of the group for their enthusiastic willingness to explain their research. The LBNL Applied Mathematics seminars were an invaluable school for learning to present my ideas rigorously yet without burying the audience under

the tons of incidentally relevant technical details. Many thanks to Ole Hald, who (in cooperation with my advisor) had taught me how this can be accomplished. Whenever my equations did not make any sense, or Taylor series refused to converge, or I was ready to rescind the basic laws of arithmetic, Ole's mere presence went a long way towards restoring my faith in the fundamental goodness (or, at the very least, the logical consistency) of the universe.

Thanks to my good friend Eugene Ingerman for his willingness to spend time on checking the craziest of my ideas, for reading this thesis so diligently and for pointing out many aspects of it which needed improvement.

Thanks to Jon Wilkening for his interest, his willingness to help and for time-sharing with me the attention of our advisor.

Lots of thanks, most certainly, to Valerie Heatlie, without whose loving care all of us would be exposed to the harsh non-mathematical realities of the outside world.

In the past years I have greatly benefited from having a chance to discuss my ideas with L.C.Evans, from whom I have learned most of what I know about the applications of functional analysis to the theory of PDEs. His perspective was instrumental in shaping my understanding of the connections to control theory.

My understanding of the relevant aspects of the PDE theory was also greatly aided by several conversations with M.G.Crandall and M.Falcone. I would like to thank them for their interest in my research.

I am indebted to my good friend and comrade-in-arms Anatoly Spitkovsky, conversations with whom have prompted me to learn much more about the role of anisotropy in wave physics. I hope that some day I'll be able to retaliate by doing something as valuable for his research in astrophysics.

My most sincere gratitude to my friend Alexander Dukhovny for invaluable mathematical conversations and for always being there when I needed his help. His devotion to mathematics has been my inspiration for many years.

Finally, I get to thank my advisor, James Sethian. For proving to me that number crunching is no substitute for geometrical intuition. For finding a way to channel my passion to doubt and question everything into something positive. For treating me as a colleague and as a friend. For always staying half a step ahead of me and for having the tactfulness to pretend then that I figured it out on my own.

Despite the tradition, I decided against thanking my family and friends for sup-

porting this research.

This research was supported by the Appl. Math. Sci. Office of Energy Research, U.S. Dept. Energy, # DE-AC03-76SF00098,and ONR, FDN00014-96-1-0381.

My family and friends, on the other hand, have supported my very existence and my notion of meaningfulness of life. To my loved ones, I owe much more than can be acknowledged in mere words.

Chapter 1

Overview

Furthermore, for the purposes of this presentation we will assume without loss of generality that $\sqrt{\pi} = \sqrt{2\pi} = 1$.

From a Berkeley introductory
lecture on Fourier series.

1.1 Notation

For the reasons of notational clarity, most of the problems considered in this thesis will be in R^2. Nevertheless, since we will be considering different numerical methods both on grids and on unstructured meshes, some notational abuse is almost unavoidable. We will do our best to adhere to the following conventions:

- the **bold fonts** will be reserved for the vectors, vector functions (e.g., $\boldsymbol{x}, \boldsymbol{y}'(t)$, etc), and for the expression of author's **uttermost excitement**;
- letters x, y, and z will refer to regular coordinate directions in R^3;
- letter X will be used to refer to both the uniform Cartesian grids and the triangulated meshes;
- letter h will denote the grid size of a uniform Cartesian grid *and* the diameter of a triangulated mesh;
- $u(\boldsymbol{x})$ will refer to the value of the solution to the PDE at the point $\boldsymbol{x}$;

- $U(\boldsymbol{x})$ will refer to the value of the numerical solution to the PDE at the point (not necessarily mesh or grid point) $\boldsymbol{x}$;
- in the discrete optimal trajectory (i.e., shortest path on a network) problems, $U(\boldsymbol{x})$ will also refer to the exact value of the solution at the node $\boldsymbol{x}$;
- $V(\boldsymbol{x})$ will refer to the *temporary solution values*, which have to be computed before obtaining the numerical solution $U(\boldsymbol{x})$;
- U_i will sometimes be used to refer to the numerical solution value at the mesh point $\boldsymbol{x_i}$;
- U_{ij} will sometimes be used to refer to the numerical solution value at the grid point $\boldsymbol{x_{ij}} = (ih, jh)$;
- u_x and u_y will be used as a shorter notation for $\frac{\partial}{\partial x}u$ and $\frac{\partial}{\partial y}u$;
- the terms **fast methods** and **single-pass methods** will be used interchangeably throughout this thesis;
- the terms **front** and **interface** will be used as synonyms in Chapter 3;
- the numerous quotes and epigraphs will have no notational significance whatsoever; they are intended to prevent the reader from paying too much attention to the mathematical subject matter.

There go my people. I must find out
where they are going so I can lead them.

Alexandre Ledru-Rollin (1807-74)
French politician

1.2 General goals

Consider the first order non-linear PDE [1]

$$G(\nabla u(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) = 0 \qquad \boldsymbol{x} \in \Omega \subset R^2 \tag{1.1}$$

with the boundary condition $u(\boldsymbol{x}) = q(\boldsymbol{x})$ given on $\partial\Omega$. In general, smoothness in boundary data does not guarantee that a smooth solution exists. For the Hamilton-Jacobi equations, weak solutions can be formally introduced; a unique *viscosity solution* can be defined using conditions on smooth test functions [16, 15]. In this thesis, we develop a family of fast methods for approximating solutions for a wide class of static Hamilton-Jacobi PDEs.

Start with a mesh X covering the domain Ω. Let $U_i = U(\boldsymbol{x_i})$ be the numerical solution at the mesh point $\boldsymbol{x_i} \in X$. Denote the set of mesh points adjacent to $\boldsymbol{x_i}$ as $N(\boldsymbol{x_i})$ and the set of values adjacent to U_i as $NU(\boldsymbol{x_i}) = \{U_j | \boldsymbol{x_j} \in N(\boldsymbol{x_i})\}$. Let $\bar{G}$ be a consistent discretization of G such that one can write

$$\bar{G}(U_i, NU(\boldsymbol{x_i}), \boldsymbol{x_i}) = 0. \tag{1.2}$$

If M is the total number of mesh points, then one needs to solve M coupled non-linear equations simultaneously. One approach is to solve this non-linear system iteratively.

Our goal in this work is to introduce a set of techniques which we label as "single-pass" algorithms. By this, we mean that we recalculate each U_i at most r times, where r depends only upon the equation 1.1 and the mesh structure, but not upon the diameter of the mesh.

1.3 Characteristics and ordering updates

To construct one-pass algorithms with efficient update orderings, we utilize the fact that the value of $u(\boldsymbol{x})$ for the first order PDE depends only on the value of u along

[1] For the sake of notational clarity we restrict our discussion to R^2 ; all results can be restated for R^n and for meshes on manifolds.

the characteristic(s) passing through the point $\boldsymbol{x}$. If $\boldsymbol{x_{i_1}}, \boldsymbol{x_{i_2}} \in N(\boldsymbol{x_i})$ are such that the characteristic for the mesh point $\boldsymbol{x_i}$ lies in the simplex $\boldsymbol{x_i x_{i_1} x_{i_2}}$, then it is useful to consider an *upwind* discretization of the PDE:

$$\bar{G}(U_i, U_{i_1}, U_{i_2}, \boldsymbol{x_i}) = 0. \tag{1.3}$$

This reduces the coupling in the system: U_i *depends* only upon U_{i_1} and U_{i_2} and not on all of the $NU(\boldsymbol{x_i})$. A recursive construction allows one to build the entire *dependency graph* for $\boldsymbol{x_i}$.

If two or more characteristics collide at the point $\boldsymbol{x}$, the solution loses smoothness. The entropy condition does not allow characteristics to be created at these collision points; hence, if $\boldsymbol{x_i}$ is far enough from these collision points, its dependency graph is actually a tree.

If the characteristic directions of the PDE were a priori known, then the dependency–ordering on the grid points would be known as well, leading to a fully de-coupled system. Formally, this construction would lead to a $O(M)$ method.

If the equation 1.1 is linear, the characteristic directions are known in advance, and such methods can be indeed implemented. An interesting example of a fast algorithm for the linear case is the "explicit finite element method" developed in 1973 for the neutron transport equation [31]. A recent (massively parallel) algorithm for solving the linear Boltzmann transport equation on unstructured grids can be found in [36].

Moreover, even if the information about the true characteristic directions is not available, an assumption can be made sometimes based on the physical interpretation of the solution. This assumption can then be used to construct an approximate version of the equation 1.1, for which a fast method can be obtained more readily. For instance, an assumption about the preferred direction can be used to build a *paraxial* version of the original PDE (see [51], for example).

In general, characteristic directions are not known in advance. Nonetheless, for some of these problems, "single-pass" methods can be devised to determine the mesh point ordering (and the characteristic directions) in the process of de-coupling the system. These Ordered Upwinding Methods will have the complexity of $O(M \log M)$. Such methods are particularly natural for the class of PDEs, for which the solution u is monotone increasing along the characteristics.

1.4 Equations, properties, perspectives.

In this work, we consider the boundary value problem for the static Hamilton-Jacobi equations of the form

$$\begin{aligned} H(\nabla u, \boldsymbol{x}) &= 1, \qquad \boldsymbol{x} \in \Omega \subset R^2, \\ u(\boldsymbol{x}) &= q(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{1.4}$$

where Hamiltonian H is assumed to be Lipschitz-continuous, convex, and homogeneous of degree 1 in the first argument:

$$H(\nabla u, \boldsymbol{x}) = \|\nabla u\| F\left(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|}\right) \tag{1.5}$$

for some function F. We will further assume that the function q is also Lipschitz-continuous, and that

$$\begin{aligned} 0 < F_1 \le F(\boldsymbol{x}, \boldsymbol{p}) \le F_2, \\ q_1 \le q(\boldsymbol{x}) \le q_2, \end{aligned}$$

for all $\boldsymbol{p}$ and $\boldsymbol{x}$.

Even for arbitrarily smooth H, q, and $\partial\Omega$, a smooth solution on Ω need not exist. In general, there are infinitely many weak Lipschitz-continuous solutions, but the unique *viscosity solution* can be defined using additional conditions on the smooth test functions [16, 15]. We begin by examining two different ways of interpreting the Hamilton-Jacobi PDE 1.4.

Chapter 2: Anisotropic min-time optimal trajectory problems.
In these problems, the speed of motion depends not only on position, but also on direction. The control-theoretic value function u is the viscosity solution of the static Hamilton-Jacobi-Bellman equation

$$\begin{aligned} &\max_{\boldsymbol{a} \in S_1} \{(\nabla u(\boldsymbol{x}) \cdot (-\boldsymbol{a})) f(\boldsymbol{x}, \boldsymbol{a})\} = 1, \quad \boldsymbol{x} \in \Omega, \\ &u(\boldsymbol{x}) = q(\boldsymbol{x}), \qquad\qquad\qquad\qquad\quad \boldsymbol{x} \in \partial\Omega. \end{aligned} \tag{1.6}$$

Here, $\boldsymbol{a}$ is the unit vector determining the direction of motion, $f(\boldsymbol{x}, \boldsymbol{a})$ is the speed of motion in the direction $\boldsymbol{a}$ starting from the point $\boldsymbol{x} \in \Omega$, and $q(\boldsymbol{x})$ is the time-penalty for exiting the domain at the point $\boldsymbol{x} \in \partial\Omega$. The maximizer $\boldsymbol{a}$ corresponds to the characteristic direction for the point $\boldsymbol{x}$.

Chapter 3: Anisotropic front expansion (contraction) problems.
In these problems, $F(\boldsymbol{n}, \boldsymbol{x})$ is interpreted as the speed of the front in the normal direction $\boldsymbol{n}$, and $\partial\Omega$ - as the initial position of the front.

$$\begin{aligned} &\|\nabla u\| F\left(\boldsymbol{x}, \tfrac{\nabla u}{\|\nabla u\|}\right) = 1, \quad && \boldsymbol{x} \in \Omega, \\ &u(\boldsymbol{x}) = 0, && \boldsymbol{x} \in \partial\Omega. \end{aligned}$$

The anisotropy is the result of F's dependence on $\boldsymbol{n}$. The level sets of the viscosity solution u correspond to the positions of the front at different times. The described numerical methods can be used only if the resulting Hamiltonian is convex.

If the speed functions F and f only depend upon their first argument, both forms of the Hamilton-Jacobi PDE reduce to the Eikonal equation

$$\|\nabla u(\boldsymbol{x})\| \; = \; K(\boldsymbol{x}), \tag{1.7}$$

where $K(\boldsymbol{x}) = \frac{1}{F(\boldsymbol{x})} = \frac{1}{f(\boldsymbol{x})}$. The Eikonal equation possesses a very useful property:
its characteristics coincide with the gradient lines of its viscosity solution u. This property is the foundation for two different "single-pass" methods for an Eikonal equation: Tsitsiklis' Algorithm (1995) and Sethian's Fast Marching Method (1996). These two methods are discussed in detail, since our new methods can be considered as a generalization of these for the anisotropic case. Tsitsiklis [53] considered the Eikonal equation primarily in the context of isotropic optimal trajectory problem and used a first-order control-theoretic upwinding discretization (section 2.2.10). Sethian [43] considered the Eikonal equation primarily in the context of isotropic front expansion problem and used a first-order finite difference upwinding discretization (section 3.4.2). In section 3.4.4 we discuss extensions of the Fast Marching Method using the second-order finite difference upwinding operators on the Cartesian grid and on unstructured meshes.

In Chapter 4, we then turn our attention to constructing fast methods for the general anisotropic case. We accomplish this by using two properties of the unique viscosity solution:

- The viscosity solution $u(\boldsymbol{x})$ is strictly increasing along the characteristics of the PDE 1.4.
- We can derive a precise upper bound on the maximum angle between the characteristic and the gradient of u.

An Ordered Upwinding Method based on a control-theoretic discretization is introduced in section 4.3. This single-pass method has a computational complexity of $O(\frac{F_2}{F_1} M \log M)$, and its convergence to the viscosity solution is proven in Chapter 5.

In Chapter 6, we explore the relationship between the two interpretations of the Hamilton-Jacobi PDE 1.4. Our analysis of the role played by anisotropy in both of these contexts is the basis for the hybrid Ordered Upwinding Methods (section 6.2). These single-pass methods are based on finite difference approximations and have the same computational complexity of $O(\frac{F_2}{F_1} M \log M)$.

Finally, we analyse the efficiency of the new methods (Chapter 7) and consider several anisotropic test problems from optimal control, computational geometry, and seismology. (Chapter 8).

Chapter 2

Control-Theoretic Perspective

> Immense power is acquired by assuring yourself in your secret reveries that you were born to control affairs.
>
> ANDREW CARNEGIE (1835 - 1919) US INDUSTRIALIST, PHILANTHROPIST

The general structure of all of the control-theoretic models is the same:

- an evolving system, whose evolution depends on some *controlled parameter*;
- the cost associated with a particular way of evolving the system, which depends, among other things, on the system's state, and on the *control* used to direct the system to that state;
- the ultimate goal of finding the way to optimally control the system (so that the resulting cost is minimized).

We will concentrate on the deterministic control problems (i.e., the behavior of the system is fully specified by the chosen control) with the *exit state termination criterion*: the process is terminated when the system reaches one of the predefined exit states, and the *exit penalty* associated with that exit state is added to the overall cost of the process.

In this chapter we discuss several well known discrete and continuous control-theoretic models. The classical discussion of many of those can be found in [9] and [8]. Our exposition is intended to provide a self-contained introduction to the principles of the

dynamic programming with the emphasis on the connections to the PDE perspective. The previously available numerical methods described here are the foundation for building our new single-pass methods (chapters 4 and 6) for solving the static Hamilton-Jacobi PDE 1.4.

2.1 Discrete optimal trajectory problem

We begin by considering the "shortest path on the network" problem extensively studied in computer science and in operations research. Numerous algorithms are available for this problem (e.g., see [3] for a catalogue of available algorithms), but Dijkstra's method [18] is probably most widely known for the specific subclass of problems in which the network is sparsely connected and all the arc-costs are positive. That method will be considered as a prototype for all the single-pass numerical methods described in this dissertation.

> More than any time in history mankind faces a crossroads.
> One path leads to despair and utter hopelessness, the other to total extinction.
> Let us pray that we have the wisdom to choose correctly.
>
> ---
>
> WOODY ALLEN, US COMEDIAN, ACTOR, FILM DIRECTOR

2.1.1 Shortest paths and value function

Consider a discrete network of nodes $X = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_M}\}$. The vehicle starts somewhere in the network and travels from node to node until it reaches one of the *exit nodes* $\boldsymbol{x} \in Q \subset X$. Thus, every vehicle's trajectory is just a finite sequence of nodes $(\boldsymbol{y_1}, \ldots, \boldsymbol{y_r})$ such that $\boldsymbol{y_k} \notin Q$ for $k < r$ and $\boldsymbol{y_r} \in Q$. There is a positive cost associated with every transition from node to node: it costs $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = K_{ij} > 0$ to pass from $\boldsymbol{x_i}$ to $\boldsymbol{x_j}$. The cost K_{ij} is assumed to be infinite if in the network there is no link from $\boldsymbol{x_i}$ to $\boldsymbol{x_j}$. For every exit node $\boldsymbol{x} \in Q$ there is a cost $q(\boldsymbol{x}) < \infty$ for exiting the network at that point. Thus, the total cost of a trajectory $(\boldsymbol{y_1}, \ldots, \boldsymbol{y_r})$ starting from the point $\boldsymbol{y_1}$ is

$$\text{Cost}(\boldsymbol{y_1}, \ldots, \boldsymbol{y_r}) = \sum_{j=1}^{r} K(\boldsymbol{y_j}, \boldsymbol{y_{j+1}}) + q(\boldsymbol{y_r}). \tag{2.1}$$

Our goal is to find the optimal (cheapest) trajectory for each node $\boldsymbol{x} \in X \backslash Q$.

Remark 2.1.1. For the purposes of our presentation, we are mostly interested in the situation, when X is a grid or (more generally) an unstructured mesh: each $\boldsymbol{x}$ is only connected to a set of its neighbors, which is assumed to be a small subset of X.

The key idea of *dynamic programming* [8, 9] is to solve this problem for all of the nodes at once and, instead of searching for a particular optimal trajectory, to derive an equation for the *value function* $U(\boldsymbol{x})$, defined as

$$\begin{cases} U(\boldsymbol{x}) = \min\limits_{\substack{\text{all the paths} \\ \text{starting at } \boldsymbol{x}}} \text{Cost}(\boldsymbol{x}, \ldots) & \boldsymbol{x} \in X \backslash Q, \\ U(\boldsymbol{x}) = q(\boldsymbol{x}) & \boldsymbol{x} \in Q. \end{cases} \tag{2.2}$$

By this definition, the value function $U(\boldsymbol{x})$ is the minimum cost for exiting the network if we have started at the node $\boldsymbol{x}$.

2.1.2 Bellman's optimality principle

Bellman's optimality principle [8] is, in effect, a way to derive such an equation for $U(\boldsymbol{x})$. Suppose, $\boldsymbol{y_1} \in X \backslash Q$. Then, by definition of the value function, there has to exist some *optimal trajectory* $(\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_r})$ such that $U(\boldsymbol{y_1}) = \text{Cost}(\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_r})$. The optimality principle merely says that if $\boldsymbol{y_2} \in X \backslash Q$, then the trajectory $(\boldsymbol{y_2}, \ldots, \boldsymbol{y_r})$ will also be optimal for the node $\boldsymbol{y_2}$. Otherwise, there would exist some other trajectory $(\boldsymbol{y_2}, \boldsymbol{z_1}, \ldots, \boldsymbol{z_p})$ such that

$$\text{Cost}(\boldsymbol{y_2}, \ldots, \boldsymbol{y_r}) > \text{Cost}(\boldsymbol{y_2}, \boldsymbol{z_1}, \ldots, \boldsymbol{z_p}),$$

and that would imply

$$U(\boldsymbol{y_1}) > \text{Cost}(\boldsymbol{y_1}, \boldsymbol{y_2}, \boldsymbol{z_1}, \ldots, \boldsymbol{z_p}),$$

which is a contradiction. If we define the set of $\boldsymbol{x}$'s neighbors

$$N(\boldsymbol{x}) = \{\boldsymbol{y} \in X \mid K(\boldsymbol{x}, \boldsymbol{y}) < \infty\},$$

then we can write Bellman's optimality principle in the form of a local equation for $U(\boldsymbol{x})$

$$U(\boldsymbol{x}) = \min_{\boldsymbol{y} \in N(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\}, \quad \text{for } \forall \boldsymbol{x} \in X \backslash Q. \tag{2.3}$$

Remark 2.1.2. Of course, we could substitute X instead of $N(\boldsymbol{x})$ and the equation would still be true. We are using this notation to emphasize that we imagine X to be some mesh and, thus, $N(\boldsymbol{x})$ is much smaller.

Equation 2.3 is non-linear and it has to hold for each node in $X \backslash Q$. Thus, if there are M such nodes, we have to solve a coupled system of M non-linear equations. Solving this entire system simultaneously is expensive; fortunately, as Dijkstra's method shows, it is also completely unnecessary.

2.1.3 Dijkstra's method

The first step towards building this efficient method is the following simple observation.

Observation 2.1.3. If $(\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_r})$ is an optimal trajectory for $\boldsymbol{y_1}$, then $U(\boldsymbol{y_1}) > U(\boldsymbol{y_2}) > \ldots > U(\boldsymbol{y_r})$.

Remark 2.1.4. This observation obviously is correct only when all the transition costs $K(\boldsymbol{x_i}, \boldsymbol{x_j})$ are positive. Some analogous property of the model will be required for all of the fast methods described in this thesis.

If we define $N_-(\boldsymbol{x}) = \{\boldsymbol{y} \in N(\boldsymbol{x}) \mid U(\boldsymbol{y}) < U(\boldsymbol{x})\}$, then the observation means that Bellman's optimality principle (2.3) can be rewritten as

$$U(\boldsymbol{x}) = \min_{\boldsymbol{y} \in N_-(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\}, \quad \text{for } \forall \boldsymbol{x} \in X \backslash Q. \tag{2.4}$$

This will essentially de-couple the system, for if $\boldsymbol{y} \in N_-(\boldsymbol{x})$, then $\boldsymbol{x} \notin N_-(\boldsymbol{y})$, and vice versa. Thus, if someone gave us the list of the nodes sorted by the value of U, we could solve the equations 2.3 one by one, producing the method with an overall complexity of $O(M)$.

Of course, this argument has an element of a trick to it: the equation 2.4 is hard to interpret, since before we know $U(\boldsymbol{y})$ and $U(\boldsymbol{x})$ it is not clear if $\boldsymbol{y} \in N_-(\boldsymbol{x})$. We cannot predict the ordering of **all** of $\boldsymbol{x} \in X \backslash Q$ before we actually compute $U(\boldsymbol{x})$. But we can find the ordering of X one node at a time:

Observation 2.1.5. Suppose that the (as-of-yet-unknown) ordering on X is $U(\boldsymbol{x_1}) \leq U(\boldsymbol{x_2}) \leq \ldots \leq U(\boldsymbol{x_M})$ and imagine that for some $i < M$ the $U(\boldsymbol{x_j})$'s are somehow known for all $j \leq i$. For every $k > i$ we define the function

$$V(\boldsymbol{x_k}) = \min_{j \leq i} \{K(\boldsymbol{x_k}, \boldsymbol{x_j}) + U(\boldsymbol{x_j})\}.$$

By the optimality principle, we observe that $V(\boldsymbol{x_k}) \geq U(\boldsymbol{x_k})$ for all $k > i$. Furthermore, if $\bar{\boldsymbol{x}}$ is such that $\forall k > i, V(\bar{\boldsymbol{x}}) \leq V(\boldsymbol{x_k})$, then

- $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$;
- if $k > i$ then $U(\bar{\boldsymbol{x}}) \leq U(\boldsymbol{x}_k)$.

Dijkstra's method [18] is based on the latter observation and works as follows:

All the nodes are divided into three classes: *Far* (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*), for which V has already been computed, but it is still unclear if $V = U$. For every *Considered* $\boldsymbol{x}$ we define the set $\mathrm{NF}(\boldsymbol{x}) = \{\boldsymbol{y} \in N(\boldsymbol{x}) \,|\, \boldsymbol{y} \text{ is } \textit{Accepted}\}$.

1. Start with all the nodes in *Far*.
2. Move the exit nodes ($\boldsymbol{y} \in Q$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the nodes $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{\boldsymbol{y} \in \mathrm{NF}(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\}. \tag{2.5}$$

4. Find the node $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).
6. Move the *Far* nodes adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$) into *Considered*.
7. Re-evaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$)

$$V(\boldsymbol{x}) := \min\{V(\boldsymbol{x}), K(\boldsymbol{x}, \bar{\boldsymbol{x}}) + U(\bar{\boldsymbol{x}})\}. \tag{2.6}$$

8. If *Considered* is not empty then go to 4.

The described algorithm has the computational complexity of $O(M \log(M))$; the factor of $\log(M)$ reflects the necessity to maintain a sorted list of the *Considered* values $V(\boldsymbol{x_i})$ to determine the next *Accepted* node[1].

[1] This variant of Dijkstra's method is often referred to as a *heap-sort Dijkstra's method* since its implementation requires a use of binary heap, d-heap, or Fibonacci heap to maintain the ordering of the *Considered* nodes efficiently[3].
The complexity estimate for the densely connected network would be $O(M^2 \log(M))$, but for our case, when X is a grid or a mesh, the precise complexity estimate is $O(rM \log(M))$, where r is the maximum number of nodes connected to a single node in X.

'Would you tell me, please, which way I ought to go from here?'
'That depends a good deal on where you want to get to,' said the Cat.
'I don't much care where–' said Alice.
'Then it doesn't matter which way you go,' said the Cat.
'–so long as I get SOMEWHERE,' Alice added as an explanation.
'Oh, you're sure to do that,' said the Cat, 'if you only walk long enough.'

CHARLES LUTWIDGE DODGSON (1832-98)
BRITISH MATHEMATICIAN, WRITER, AND POET

2.2 Continuous optimal trajectory problem

Consider a problem of finding an optimal trajectory for a vehicle moving with a unit speed in the domain $\Omega \subset R^2$:

$$\begin{aligned} \boldsymbol{y}'(t) &= \boldsymbol{a}(t), \\ \boldsymbol{y}(0) &= \boldsymbol{x} \in \Omega, \end{aligned} \tag{2.7}$$

where $\boldsymbol{y}(t)$ is the position of the vehicle at time t, $S_1 = \{\boldsymbol{a} \in R^2 | \ \|\boldsymbol{a}\| = 1\}$ is the set of *admissible control values*, and $\mathcal{A} = \{\boldsymbol{a} : R_{+,0} \mapsto S_1 \mid \boldsymbol{a}(\cdot) \text{ is measurable}\}$ is the set of *admissible controls*.

We are interested in studying $\boldsymbol{y}(t)$ only while the vehicle remains inside Ω, i.e., until the exit time

$$T(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \inf\{t \in R_{+,0} | \boldsymbol{y}(t) \in \partial\Omega\}.$$

Given a running cost function $K : (\Omega \times S_1) \mapsto R_+$ and an exit cost $q : \partial\Omega \mapsto R_{+,0}$, we can define the total cost of using control $\boldsymbol{a}(\cdot)$ starting from the point $\boldsymbol{x} \in \Omega$:

$$\text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \int_0^{T(\boldsymbol{x}, \boldsymbol{a}(\cdot))} K(\boldsymbol{y}(s), \boldsymbol{a}(s)) ds \quad + \quad q(\boldsymbol{y}(T(\boldsymbol{x}, \boldsymbol{a}(\cdot)))). \tag{2.8}$$

Unless otherwise explicitly specified, we will assume in all of the following chapters that both K and q are Lipschitz-continuous and that there exist constants K_1, K_2, q_1, q_2 such that

$$\begin{aligned} 0 < K_1 \leq K(\boldsymbol{x}, \boldsymbol{a}) \leq K_2 < \infty \qquad & \text{for } \forall \boldsymbol{x} \in \Omega \text{ and } \forall \boldsymbol{a} \in S_1; \\ 0 < q_1 \leq q(\boldsymbol{x}) \leq q_2 < \infty \qquad & \text{for } \forall \boldsymbol{x} \in \partial\Omega. \end{aligned} \tag{2.9}$$

Remark 2.2.1.

1. $\boldsymbol{K_1 > 0}$ is a "no free lunch" assumption: it always costs something to go in any direction.

2. $\boldsymbol{K_2 < \infty}$ means that there are no infinitely expensive states of the system or directions of motion.

3. We will see that the assumptions on the bounds of the running cost K are fundamentally important for the fast methods under consideration.

4. The computational complexity and the speed of convergence of the general fast methods (chapters 4 and 6) will depend upon the *anisotropy coefficient* $\Upsilon \equiv \frac{K_2}{K_1}$.

As in the discrete case, the key idea of dynamic programming [8] is to define the value function $u(\boldsymbol{x})$ such that

$$\begin{cases} u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)} \text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) & \boldsymbol{x} \in \Omega \backslash \partial\Omega, \\ u(\boldsymbol{x}) = q(\boldsymbol{x}) & \boldsymbol{x} \in \partial\Omega, \end{cases} \tag{2.10}$$

Remark 2.2.2.

1. The infimum in the above definition is due to the fact that there are infinitely many possible trajectories and the optimal control $\boldsymbol{a}(\cdot)$ in general does not have to exist; therefore, when proving properties of the value function u we should instead resort to using the *ϵ-suboptimal controls $\boldsymbol{a}(\cdot)$ such that*

$$\text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) \; < \; u(\boldsymbol{x}) + \epsilon. \tag{2.11}$$

   By the definition of the value function, such an ϵ-suboptimal control exists for all $\boldsymbol{x} \in \Omega$ and $\epsilon > 0$.

2. The Lipschitz-continuity of K and q is not really needed to define the value function. Most of the properties shown for u in the next two sections still hold, but the connection to Hamilton-Jacobi-Bellman PDEs (section 2.2.3) becomes problematic.

> The art of doing mathematics consists in finding that special case which contains all the germs of generality.
>
> DAVID HILBERT (1862-1943)

Example 2.2.3. Consider the problem, in which $K(\boldsymbol{x}, \boldsymbol{a}) = 1$ for all $\boldsymbol{x} \in \Omega \backslash \partial\Omega$, $\boldsymbol{a} \in S_1$, and $q(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \partial\Omega$. In that case $\text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = T(\boldsymbol{x}, \boldsymbol{a}(\cdot))$ and, therefore, $u(\boldsymbol{x})$ is the length of the shortest path from $\boldsymbol{x}$ to the boundary $\partial\Omega$.

As we will quickly show, this very simple special case most definitely does not contain "all the germs of generality", but it is, nevertheless, a useful test problem for the initial testing of the numerical methods. Solving this simple problem on manifolds will also prove a useful alternative to solving much more complicated equations in the plane (see section 3.5).

Some of the properties of the general problem can be seen even in this simple case; for instance, Figure 2.1 shows that the value function does not have to be smooth even when K , q and $\partial\Omega$ are perfectly "nice"; nor does the optimal trajectory have to be unique.

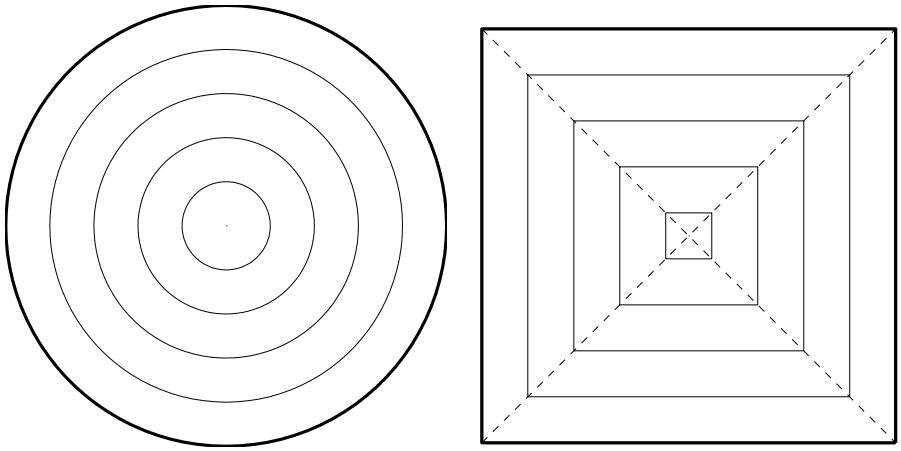


Figure 2.1: The "distance from the boundary" equation: $\|\nabla u(\boldsymbol{x})\| = 1$ on Ω, $u(\boldsymbol{x}) = 0$ on $\partial\Omega$. Level sets of the distance from the boundary function for two different domains. The shocks (dashed lines on the figure) are the collections of points, for which the closest point on the boundary is not unique.

2.2.1 Bellman's optimality principle.

The optimality principle for the continuous case is quite similar to what we already have seen in section 2.1.2.

Lemma 2.2.4 (Optimality Principle). *Let $d(\boldsymbol{x})$ be the minimum distance to the boundary $\partial\Omega$. Then for every point $\boldsymbol{x} \in \Omega\backslash\partial\Omega$ and for any $\tau < d(\boldsymbol{x})$,*

$$u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)} \left\{ \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \; + \; u(\boldsymbol{y}(\tau)) \right\}. \tag{2.12}$$

Proof. Choose an arbitrary $\boldsymbol{x} \in \Omega\backslash\partial\Omega$ and $\tau < d(\boldsymbol{x})$. This choice of τ ensures that if $\boldsymbol{y}(0) = \boldsymbol{x}$ then $\boldsymbol{y}(\tau) \in \Omega\backslash\partial\Omega$ for any choice of control $\boldsymbol{a}(\cdot)$. Suppose $\boldsymbol{a_1}(\cdot)$ is an ϵ-suboptimal control for $\boldsymbol{x}$, and let $\boldsymbol{x_1} = \boldsymbol{y}(\tau)$.

Claim: $\boldsymbol{b}(t) = \boldsymbol{a_1}(t + \tau)$ is an ϵ-suboptimal control for $\boldsymbol{x_1}$. Indeed, if that were not the case, then there would exist some control $\boldsymbol{b_1}(\cdot)$ such that

$$\text{Cost}(\boldsymbol{x_1}, \boldsymbol{b_1}(\cdot)) + \epsilon < \text{Cost}(\boldsymbol{x_1}, \boldsymbol{b}(\cdot)),$$

which would imply that for the control

$$\boldsymbol{a_2}(t) = \begin{cases} \boldsymbol{a_1}(t) & t \in [0, \tau), \\ \boldsymbol{b_1}(t - \tau) & t \geq \tau \end{cases}$$

the cost $\text{Cost}(\boldsymbol{x}, \boldsymbol{a_2}(\cdot)) < \text{Cost}(\boldsymbol{x}, \boldsymbol{a_1}(\cdot)) - \epsilon \leq u(\boldsymbol{x})$, which is impossible. Thus,

$$u(\boldsymbol{x}) + \epsilon \geq \text{Cost}(\boldsymbol{x}, \boldsymbol{a_1}(\cdot)) \geq \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a_1}(s))ds + u(\boldsymbol{x_1}),$$

where $\boldsymbol{a_1}(\cdot)$ and the corresponding $\boldsymbol{x_1}$ can be found for an arbitrarily small ϵ. So,

$$u(\boldsymbol{x}) \geq \inf_{\boldsymbol{a}(\cdot)} \left\{ \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \; + \; u(\boldsymbol{y}(\tau)) \right\}.$$

The remaining half of the proof is even easier.

Choose an arbitrary control $\boldsymbol{a_1}(\cdot)$ set $\boldsymbol{y}(0) = \boldsymbol{x}$ and let $\boldsymbol{x_1} = \boldsymbol{y}(\tau)$. Choose an ϵ-suboptimal control $\boldsymbol{b}(\cdot)$ for the point $\boldsymbol{x_1}$. Define the control

$$\boldsymbol{a_2}(t) = \begin{cases} \boldsymbol{a_1}(t) & t \in [0, \tau), \\ \boldsymbol{b}(t - \tau) & t \geq \tau, \end{cases}$$

and observe that

$$\begin{aligned} u(\boldsymbol{x}) \leq \mathrm{Cost}(\boldsymbol{x}, \boldsymbol{a_2}(\cdot)) &= \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds + \mathrm{Cost}(\boldsymbol{x_1}, \boldsymbol{b}(\cdot)) \leq \\ &\leq \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds + u(\boldsymbol{x_1}) + \epsilon. \end{aligned}$$

Since $\boldsymbol{b}(\cdot)$ can be chosen so that the above holds for an arbitrarily small ϵ, we see that

$$u(\boldsymbol{x}) \leq \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \ + \ u(\boldsymbol{y}(\tau)),$$

for any control $\boldsymbol{a}(\cdot)$. □

Remark 2.2.5. An analogous optimality principle can be proven without restricting τ to be less than $d(\boldsymbol{x})$. The infimum, however, would be taken over the set of all controls $\boldsymbol{a}(\cdot)$ such that $\boldsymbol{y}(\tau)$ is still inside Ω.

Remark 2.2.6. To simplify the presentation in the rest of this chapter, we will somewhat loosely refer to the optimal trajectories (i.e., trajectories corresponding to the optimal controls). If such optimal controls do not exist, the same (or ever-so-slightly more carefully stated) properties can be formulated and proven for the ϵ-suboptimal controls and trajectories (as in the above proof).

2.2.2 Properties of the value function.

Several properties of the value function immediately follow from the definition and from the optimality principle.

Lemma 2.2.7 (Lipschitz-continuity). *If $\bar{\Omega}$ is convex and $\boldsymbol{x_1}, \boldsymbol{x_2} \in \Omega \backslash \partial\Omega$ then*

$$|u(\boldsymbol{x_1}) - u(\boldsymbol{x_2})| \leq K_2 \|\boldsymbol{x_1} - \boldsymbol{x_2}\|.$$

Proof. Let $\tau = \|\boldsymbol{x_1} - \boldsymbol{x_2}\|$ and consider the control $\boldsymbol{a}(t) = \frac{\boldsymbol{x_2} - \boldsymbol{x_1}}{\tau}$. By the optimality principle,

$$u(\boldsymbol{x_1}) \leq \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \ + \ u(\boldsymbol{x_2}) \leq K_2 \|\boldsymbol{x_1} - \boldsymbol{x_2}\| \ + \ u(\boldsymbol{x_2}).$$

Similarly, we can show that

$$u(\boldsymbol{x_2}) \leq \int_0^\tau K(\boldsymbol{y}(s), -\boldsymbol{a}(s))ds \ + \ u(\boldsymbol{x_1}) \leq K_2 \|\boldsymbol{x_1} - \boldsymbol{x_2}\| \ + \ u(\boldsymbol{x_1}).$$

□

"The shortest path between two points is always under construction."

Anonymous Wisdom

Remark 2.2.8. The convexity of $\bar{\Omega}$ is required to ensure that the straight line segment $\boldsymbol{x_1 x_2}$ lies inside Ω and, thus, is a valid trajectory. A similar statement can be proven without the convexity assumption if $\boldsymbol{x_1}, \boldsymbol{x_2}$ are far enough from $\partial\Omega$ or if $\|\boldsymbol{x_1} - \boldsymbol{x_2}\|$ is replaced by the length of the shortest path from $\boldsymbol{x_1}$ to $\boldsymbol{x_2}$ inside the domain.

Lemma 2.2.9 (Boundedness). *If $\boldsymbol{x} \in \Omega \backslash \partial\Omega$ then*

$$K_1 d(\boldsymbol{x}) + q_1 \leq u(\boldsymbol{x}) \leq K_2 d(\boldsymbol{x}) + q_2.$$

Proof. Let $\tilde{\boldsymbol{x}}$ be a point on $\partial\Omega$ such that $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| = d(\boldsymbol{x})$. Let $\tau = \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|$ and consider the control $\boldsymbol{a}(t) = \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau}$. By the definition of the value function,

$$u(\boldsymbol{x}) \leq \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \ + \ u(\tilde{\boldsymbol{x}}) \leq d(\boldsymbol{x})K_2 + q_2.$$

On the other hand, if $\boldsymbol{a_1}(\cdot)$ is an optimal control for $\boldsymbol{x}$, then

$$u(\boldsymbol{x}) = \int_0^{T(\boldsymbol{x}, \boldsymbol{a_1}(\cdot))} K(\boldsymbol{y}(s), \boldsymbol{a_1}(s))ds \ + \ u(\boldsymbol{y}(T(\boldsymbol{x}, \boldsymbol{a_1}(\cdot)))) \geq K_1 d(\boldsymbol{x}) + q_1.$$

□

Lemma 2.2.10. *If $\boldsymbol{y}(t) = \boldsymbol{a}(t)$ is an optimal trajectory for a point $\boldsymbol{x}$ (i.e., $\boldsymbol{y}(0) = \boldsymbol{x}$ and $u(\boldsymbol{x}) = \text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot))$), then the function $u(\boldsymbol{y}(t))$ is strictly decreasing for $t \in [0, T(\boldsymbol{x}, \boldsymbol{a}(\cdot))]$.*

Proof. Follows directly from the proof of the optimality criterion and the fact that $K(\boldsymbol{x}, \boldsymbol{a}) > K_1 > 0$. □

Lemma 2.2.11. *Consider a point $\bar{\boldsymbol{x}} \in \Omega \backslash \partial\Omega$. Then, for any constant C such that $q_2 \leq C \leq u(\bar{\boldsymbol{x}})$, the optimal trajectory for $\bar{\boldsymbol{x}}$ will intersect the level set $u(\boldsymbol{x}) = C$ at some point $\tilde{\boldsymbol{x}}$. If $\bar{\boldsymbol{x}}$ is distance d_1 away from that level set, then*

$$\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq d_1 \frac{K_2}{K_1}. \tag{2.13}$$

In America, it's not how much an item costs, it's how much you save.

Provably Optimal Trajectories in USA & Abroad, paperback edition.

Proof. Let $\boldsymbol{a}(\cdot)$ be an optimal control for $\bar{\boldsymbol{x}}$. The intersection point $\tilde{\boldsymbol{x}} = \boldsymbol{y}(\tau)$ exists because of the continuity of the value function and of the optimal trajectory: $u(\bar{\boldsymbol{x}}) \geq C \geq q_2 \geq u(y(T(\bar{\boldsymbol{x}}, \boldsymbol{a}(\cdot))))$.

Therefore,

$$u(\bar{\boldsymbol{x}}) = \int_0^{\tau} K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \ + \ u(\tilde{\boldsymbol{x}}) \geq \|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|K_1 + C.$$

There also exists some point $\hat{\boldsymbol{x}}$ on the level set such that $\|\bar{\boldsymbol{x}} - \hat{\boldsymbol{x}}\| = d_1$. Consider a control $\boldsymbol{a_1}(t) = \frac{\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}}{d_1}$. By the optimality principle,

$$u(\bar{\boldsymbol{x}}) \leq \int_0^{d_1} K(\boldsymbol{y}(s), \boldsymbol{a_1}(s))ds \ + \ u(\hat{\boldsymbol{x}}) \leq d_1 K_2 + C.$$

Thus, $\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq d_1 \frac{K_2}{K_1}$. □

Lemma 2.2.12. *Consider an unstructured (triangulated) mesh X of diameter h on Ω. Consider a simple closed curve $\Gamma \subset \Omega \backslash \partial\Omega$ with the property that for any point $\boldsymbol{x}$ on Γ, there exists a mesh point $\hat{\boldsymbol{x}}$ inside Γ such that $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| < h$. Suppose a mesh point $\bar{\boldsymbol{x}}$ is such that $u(\bar{\boldsymbol{x}}) \leq u(\boldsymbol{x_i})$ for all the mesh points $\boldsymbol{x_i} \in X$ inside Γ. The optimal trajectory for $\bar{\boldsymbol{x}}$ will intersect Γ at some point $\tilde{\boldsymbol{x}}$ such that*

$$\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq h \frac{K_2}{K_1}. \tag{2.14}$$

Remark 2.2.13. The condition "for $\forall \boldsymbol{x} \in \Gamma$ there exists some nearby mesh point $\hat{\boldsymbol{x}}$ inside Γ" is the requirement that the curve Γ has to be "well-resolved" by X.

Proof. Let $\boldsymbol{a}(\cdot)$ be an optimal control for $\bar{\boldsymbol{x}}$. The intersection point $\tilde{\boldsymbol{x}} = \boldsymbol{y}(\tau)$ exists because of the continuity of Γ and of the optimal trajectory. Therefore,

$$u(\bar{\boldsymbol{x}}) = \int_0^{\tau} K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds \ + \ u(\tilde{\boldsymbol{x}}) \geq \|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|K_1 + \ u(\tilde{\boldsymbol{x}}).$$

Let $\hat{\boldsymbol{x}}$ be a mesh point such that $\|\tilde{\boldsymbol{x}} - \hat{\boldsymbol{x}}\| = \tau_1 \leq h$. Consider a control $\boldsymbol{a_1}(t) = \frac{\tilde{\boldsymbol{x}} - \hat{\boldsymbol{x}}}{\tau_1}$. Then, by the optimality principle,

$$u(\hat{\boldsymbol{x}}) \leq \int_0^{\tau_1} K(\boldsymbol{y}(s), \boldsymbol{a_1}(s))ds \ + \ u(\tilde{\boldsymbol{x}}) \leq hK_2 + \ u(\tilde{\boldsymbol{x}}).$$

To complete the proof, we recall that $u(\bar{\boldsymbol{x}}) \leq u(\hat{\boldsymbol{x}})$. □

2.2.3 Hamilton-Jacobi-Bellman PDE.

By analogy with the discrete case, we would now like to use Bellman's optimality principle to derive an equation for the value function. The following "naive" derivation will be based on the assumption that $u(\boldsymbol{x})$ is smooth in the neighborhood of $\boldsymbol{x}$. We know already that generally this does not have to be the case (see Figure 2.1), but we will proceed with this formal derivation anyway.

For some small τ we can rewrite the optimality principle as follows:

$$\begin{aligned} u(\boldsymbol{x}) &= \inf_{\boldsymbol{a}(\cdot)} \left\{ \int_0^\tau K(\boldsymbol{y}(s), \boldsymbol{a}(s)) ds \ + \ u(\boldsymbol{y}(\tau)) \right\} \\ &= \inf_{\boldsymbol{a}(\cdot)} \left\{ \int_0^\tau K\left(\boldsymbol{y}(0) + O(\tau), \boldsymbol{a}(0) + O(\tau)\right) ds \ + \ u(\boldsymbol{x} + \tau \boldsymbol{a}(0) + O(\tau^2)) \right\} \\ &= \min_{\boldsymbol{a} \in S_1} \left\{ \tau K(\boldsymbol{x}, \boldsymbol{a}) \ + \ u(\boldsymbol{x} + \tau \boldsymbol{a}) + O(\tau^2) \right\} \\ &= \min_{\boldsymbol{a} \in S_1} \tau \left\{ K(\boldsymbol{x}, \boldsymbol{a}) \ + \ (\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}) + O(\tau) \right\} \ + \ u(\boldsymbol{x}). \end{aligned}$$

Remark 2.2.14. Note that the infimum was replaced by the minimum as we switched from $\boldsymbol{a}(\cdot) \in \mathcal{A}$ to $\boldsymbol{a} \in S_1$. This is due to the fact that the optimal control/trajectory might not exist, but the locally optimal control value/direction is always well defined.

Since the above should formally hold for all small τ, we see that $u(\boldsymbol{x})$ should satisfy the following PDE:

$$\begin{gathered} \min_{\boldsymbol{a} \in S_1} \{ K(\boldsymbol{x}, \boldsymbol{a}) + \nabla u(\boldsymbol{x}) \cdot \boldsymbol{a} \} = 0, \ \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = q(\boldsymbol{x}), \ \boldsymbol{x} \in \partial\Omega. \end{gathered} \tag{2.15}$$

This equation is known as a *Hamilton-Jacobi-Bellman PDE*. If we denote the Hamiltonian $H(\boldsymbol{p}, \boldsymbol{x}) = \min_{\boldsymbol{a} \in S_1} \{ K(\boldsymbol{x}, \boldsymbol{a}) + \boldsymbol{p} \cdot \boldsymbol{a} \} + 1$, we can write it in the form $H(\nabla u, \boldsymbol{x}) = 1$, which was used in section 1.4. We observe, however, that

- H is not convex in the first argument (it is actually concave);
- H is not homogeneous of degree one in the first argument.

This H does not seem to be similar to the class of Hamiltonians specified in section 1.4, but in section 2.2.5 we will see that it is equivalent to another Hamilton-Jacobi PDE, which belongs to that class.

Remark 2.2.15. If we formally solve the characteristic ODEs for the equation 2.15, we will find that the direction of the characteristic $\boldsymbol{a} \in S_1$ for the point $\boldsymbol{x} \in \Omega \backslash \partial\Omega$ is, in fact,

the minimizer and $K(\boldsymbol{x}, \boldsymbol{a}) + \nabla u(\boldsymbol{x}) \cdot \boldsymbol{a} \ = 0$. Thus, the characteristic directions of the PDE 2.15 are exactly the optimal control values and the characteristic lines are the optimal trajectories for the control problem.

> No man means all he says, and yet very few say all they mean, for words are slippery and thought is **viscous**.
>
> HENRY BROOKS ADAMS (1838 - 1918)
> US HISTORIAN, AUTHOR

The smooth solution to equation 2.15 might not exist even for smooth K, q, and $\partial\Omega$. Generally it admits infinitely many weak solutions, but the unique *viscosity solution* can be defined using the following conditions on smooth test functions [16, 15].

A bounded, uniformly continuous function u is the viscosity solution of equation (2.15) if the following holds for each $\phi \in C_c^\infty(\Omega)$:

(i) if $u - \phi$ has a local maximum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla \phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} \geq 0; \tag{2.16}$$

(ii) if $u - \phi$ has a local minimum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla \phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} \leq 0; \tag{2.17}$$

Remark 2.2.16. As proven in [16, 15], there exists a unique viscosity solution to the equation 2.15. That solution is Lipschitz-continuous and, therefore, differentiable almost everywhere in Ω (see [19], for example). If $\nabla u(\boldsymbol{x_0})$ is defined then the function is also a "classical solution" of the PDE, namely

$$\min_{\boldsymbol{a} \in S_1} \{K(\boldsymbol{x_0}, \boldsymbol{a}) + \nabla u(\boldsymbol{x_0}) \cdot \boldsymbol{a}\} = 0.$$

As proven in [16, 15, 14, 12][2], Bellman's optimality principle can be used to rigorously show that the value function u of the optimal trajectory problem satisfies the above test conditions and is the unique *viscosity solution* of the equation 2.15. We will use the above definition to prove the convergence of our general method in Chapter 5.

[2]The control-theoretic problems discussed in these papers are slightly different. They consider infinite horizon or exit time problems with the time discounted running costs, e.g., $\text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \int_0^\infty e^{-\lambda s} K(\boldsymbol{y}(s), \boldsymbol{a}(s)) ds$. Thus, the resulting PDE is also slightly different, but Kruzhkov's transform can be used to obtain the mapping from one to another. See [14], [4], for example.

Remark 2.2.17. For all the technical glamour, the above test conditions are supposed to regulate how the solution behaves at the *shocks* (wherever ∇u is undefined). Whenever ∇u is well defined, the local max/min conditions would force the test function to have $\nabla\phi = \nabla u$; thus, the equality would be attained in the test inequalities 2.17 and 2.16. These inequalities specify a criterion similar to the *entropy condition* for the hyperbolic conservation laws: the characteristics can run into shock, but characteristics never emanate from the shock (see [19], for example).
The term *viscosity solution* refers to the fact that u can also be obtained by the method of *vanishing viscosity*, as a uniform limit:

$$u(\boldsymbol{x}) = \lim_{\varepsilon \to 0} u_\varepsilon(\boldsymbol{x}),$$

where u_ε is the smooth and unique solution of the regularized equation

$$H(\nabla u_\varepsilon(\boldsymbol{x}), \boldsymbol{x}) = 1 + \epsilon \Delta u_\varepsilon(\boldsymbol{x}).$$

Example 2.2.18 (Defining viscosity solution for the distance function). The smooth test functions define which one of the weak Lipschitz-continuous solutions on Figure 2.2 should be chosen as the viscosity solution of the equation $|u'(x)| = 1$ with the boundary conditions $u(0) = u(1) = 0$. (From the control-theoretic perspective the answer is obvious since we would want to recover the distance function from the boundary on the interval $[0, 1]$; see the derivation in section 2.2.4.)

2.2.4 Eikonal equation: the glory of isotropy.

We will now consider a very important special case, in which the running cost function K is isotropic, i.e., depends only on the position of the vehicle. In that case,

$$\begin{aligned} 0 &= \min_{\boldsymbol{a} \in S_1} \{K(\boldsymbol{x}) + \nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}\} \\ &= K(\boldsymbol{x}) + \min_{\boldsymbol{a} \in S_1} \{\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}\} \\ &= K(\boldsymbol{x}) - \|\nabla u(\boldsymbol{x})\| \end{aligned}$$

We have derived the Eikonal equation:

$$\|\nabla u(\boldsymbol{x})\| = K(\boldsymbol{x}),$$

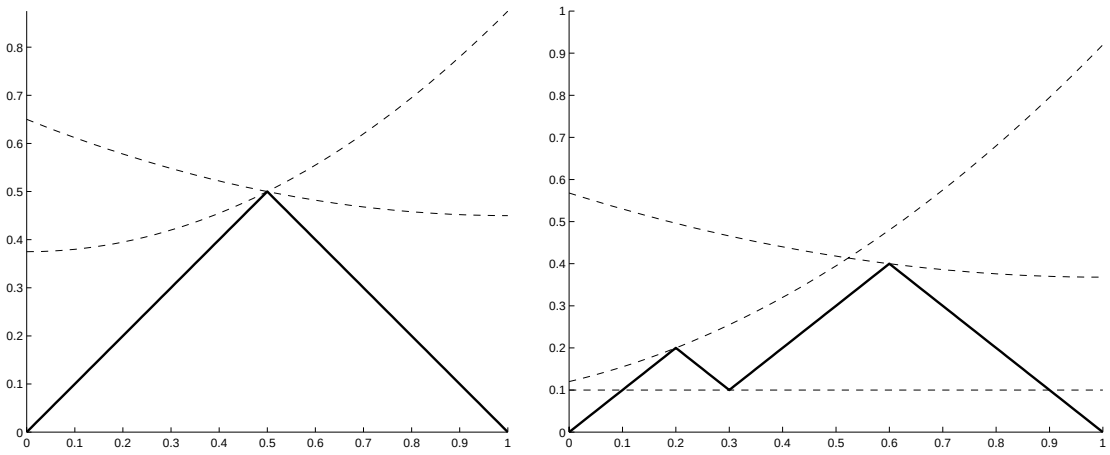


Figure 2.2: Two weak solutions to the distance equation on an interval: $|u'(x)| = 1$ for $x \in (0,1)$, and $u(0) = u(1) = 0$. Smooth test functions (dotted lines) are used to determine the viscosity solution. The second weak solution fails the test at the point $x_0 = 0.3$: if $\phi(x) = .1$ is the test function, then $(u - \phi)(x)$ has a strict local minimum at x_0, and the inequality 2.17 requires $|\phi'(x_0)| \geq 1$.

which is a significantly more general class than our previous example: the distance function is obtained when $K(\boldsymbol{x}) = 1$. The Eikonal equation describes a multitude of isotropic processes and has applications in geometric optics, computational geometry, robotic navigation, photolithography, computer vision, seismology and many other application domains.

We would like to emphasize one particular property of the Eikonal equations. In the above derivation, the minimizer $\boldsymbol{a} = \frac{-\nabla u}{\|\nabla u\|}$. Thus, the gradient lines coincide with the characteristics.

Property 2.2.19 (Causality). *If $\nabla u(\boldsymbol{x})$ is defined and $\boldsymbol{x}\boldsymbol{x_1}\boldsymbol{x_2}$ is a sufficiently small acute simplex, which contains the characteristic for $\boldsymbol{x}$, then $u(\boldsymbol{x}) \geq \max\{u(\boldsymbol{x_1}), u(\boldsymbol{x_2})\}$.*

This is the key property for understanding and enhancing the fast methods described in sections 2.2.10 and 3.4.2. Each of these two methods is based on an observation that some numerical scheme mimics the above property of the PDE.

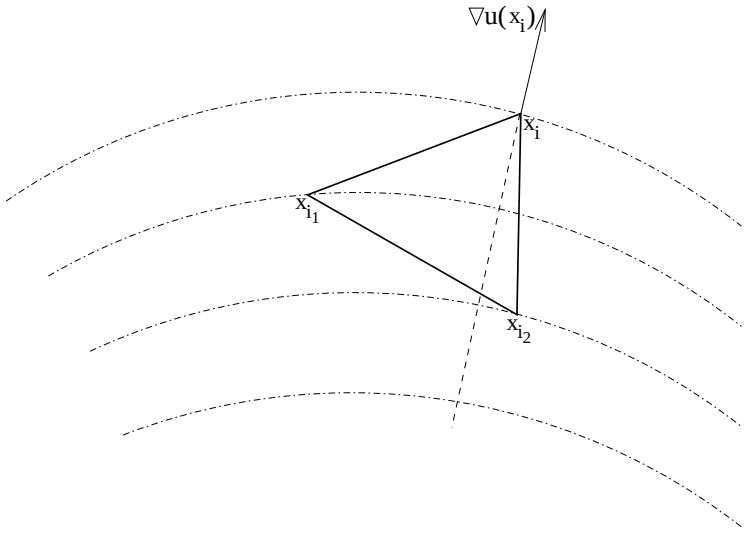


Figure 2.3: The characteristics and the gradient lines are the same for the Eikonal equation. If the characteristic lies in the simplex $\boldsymbol{x_i x_{i_1} x_{i_2}}$ then the gradient $\nabla u(\boldsymbol{x_i})$ points from that simplex. The dotted line shows the characteristic passing through the point $\boldsymbol{x_i}$, and the dash-dotted lines show the level sets of the viscosity solution u.

While we are postponing life speeds up.

LUCIUS ANNAEUS SENECA (4BC-65AD)
ROMAN PHILOSOPHER, STATESMAN

2.2.5 Equivalent control problem: constant cost vs. constant speed

So far we have described a problem of finding an optimal trajectory for a vehicle moving with a unit speed inside the domain Ω (we have allowed the vehicle's running cost K to depend upon the choice of the direction and upon the current location in the domain). We will now show that this problem is in some sense equivalent to finding an optimal trajectory for a vehicle moving in the same domain with changing speed, but constant running cost. The idea is to re-parameterize each trajectory so that the new speed is inversely proportional to the running cost.

Consider a vehicle such that its position in Ω is given by

$$\boldsymbol{z}'(r) = f(\boldsymbol{a}(t))\boldsymbol{a}(t) = \frac{\boldsymbol{a}(t)}{K(\boldsymbol{y}(t), \boldsymbol{a}(t))}, \tag{2.18}$$

$$\boldsymbol{z}(0) = \boldsymbol{x} \quad \in \Omega, \tag{2.19}$$

where $r = r(t) = \int_0^t K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds$. Then

$$\frac{d}{dt}(\boldsymbol{z}(r(t)) - \boldsymbol{y}(t)) = \boldsymbol{z}'(r(t)) \cdot r'(t) - \boldsymbol{y}'(t) = \frac{\boldsymbol{a}(t)}{K(\boldsymbol{y}(t), \boldsymbol{a}(t))} \cdot K(\boldsymbol{y}(t), \boldsymbol{a}(t)) - \boldsymbol{a}(t) = 0. \quad (2.20)$$

Thus, $\boldsymbol{y}(t) = \boldsymbol{z}(r)$. We note that $r'(t)$ is a strictly positive function; thus, $r(t)$ is invertible, and we can rewrite equation (2.18) as

$$\boldsymbol{z}'(r) = f(\bar{\boldsymbol{a}}(r))\bar{\boldsymbol{a}}(r) = \frac{\bar{\boldsymbol{a}}(r)}{K(\boldsymbol{z}(r), \bar{\boldsymbol{a}}(r))}, \quad (2.21)$$

where $\bar{\boldsymbol{a}}(\cdot)$ is such that $\bar{\boldsymbol{a}}(r) = \boldsymbol{a}(t(r))$. If we consider the latter problem with running cost equal 1 and with the same exit cost q, we see that the cost of using the control $\bar{\boldsymbol{a}}()$ is

$$\begin{aligned} \text{Cost}_1(\boldsymbol{x}, \bar{\boldsymbol{a}}(\cdot)) &= \int_0^{r(T(\boldsymbol{x}, \boldsymbol{a}(\cdot)))} ds \quad + \quad q(\boldsymbol{z}(r(T(\boldsymbol{x}, \boldsymbol{a}(\cdot))))) \\ &= r(T(\boldsymbol{x}, \boldsymbol{a}(\cdot))) + q(\boldsymbol{y}(T(\boldsymbol{x}, \boldsymbol{a}(\cdot)))) \\ &= \int_0^{T(\boldsymbol{x}, \boldsymbol{a}(\cdot))} K(\boldsymbol{y}(s), \boldsymbol{a}(s))ds + q(\boldsymbol{y}(T(\boldsymbol{x}, \boldsymbol{a}(\cdot)))) \\ &= \text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)). \end{aligned} \quad (2.22)$$

We see that

$$u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot) \in \mathcal{A}} \text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \inf_{\bar{\boldsymbol{a}}(\cdot) \in \mathcal{A}} \text{Cost}_1(\boldsymbol{x}, \bar{\boldsymbol{a}}(\cdot)), \quad (2.23)$$

i.e., that $u(\boldsymbol{x})$ is also the value function corresponding to the control problem (2.21). As such it has to be the unique viscosity solution of the corresponding Hamilton–Jacobi–Bellman equation

$$\min_{\boldsymbol{a} \in S_1} \{\nabla u(\boldsymbol{x}) \cdot \frac{\boldsymbol{a}}{K(\boldsymbol{x}, \boldsymbol{a})}\} + 1 = \min_{\boldsymbol{a} \in S_1} \{(\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}) f(\boldsymbol{x}, \boldsymbol{a})\} + 1 = 0, \qquad \boldsymbol{x} \in \Omega, \quad (2.24)$$

with the same boundary condition $u(\boldsymbol{x}) = q(\boldsymbol{x})$ for $\boldsymbol{x} \in \partial\Omega$.

Remark 2.2.20. Note that if we choose a Hamiltonian $H = -\min_{\boldsymbol{a} \in S_1}\{(\boldsymbol{p} \cdot \boldsymbol{a}) f(\boldsymbol{x}, \boldsymbol{a})\} = \max_{\boldsymbol{a} \in S_1}\{(\boldsymbol{p} \cdot (-\boldsymbol{a})) f(\boldsymbol{x}, \boldsymbol{a})\}$, then the above Hamilton-Jacobi-Bellman equation can be rewritten in the familiar form $H(\nabla u, \boldsymbol{x}) = 1$. Moreover, this Hamiltonian is convex and homogeneous of degree one in the first argument; thus, it belongs to the class of problems described in section 1.4.

Remark 2.2.21. Since the equations 2.15 and 2.24 were shown to be equivalent, we will switch to whichever language is more convenient for the particular application. By definition $f(\boldsymbol{x}, \boldsymbol{a}) = \frac{1}{K(\boldsymbol{x},\boldsymbol{a})}$; thus, we know that $0 < K_2^{-1} = f_1 < f < f_2 = K_1^{-1}$ and the anisotropy coefficient $\Upsilon = \frac{K_2}{K_1} = \frac{f_2}{f_1}$.

Remark 2.2.22. The control-theoretic problem defined by the equations 2.21 and 2.24 is often referred to as a *min-time optimal trajectory problem*, since the total running cost associated with a particular control is precisely the time it takes to trace the corresponding trajectory.

2.2.6 Modified definition of viscosity solution

According to the traditional definition (see [15]), a bounded, uniformly continuous function u is the viscosity solution of equation (2.24) if the following holds for each $\phi \in C_c^\infty(\Omega)$:

(i) if $u - \phi$ has a local maximum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{a}}{K(\boldsymbol{x_0}, \boldsymbol{a})}\} + 1 \geq 0; \tag{2.25}$$

(ii) if $u - \phi$ has a local minimum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{a}}{K(\boldsymbol{x_0}, \boldsymbol{a})}\} + 1 \leq 0; \tag{2.26}$$

Define $S_1^{\phi,\boldsymbol{x}} = \{\boldsymbol{a} \in S_1 \mid \boldsymbol{a} \cdot \nabla\phi(\boldsymbol{x}) \leq -\|\nabla\phi(\boldsymbol{x})\|\frac{K_1}{K_2}\}$. It is easy to see that using $S_1^{\phi,\boldsymbol{x_0}}$ instead of S_1 in the inequalities (2.26) and (2.25) above, we will get an equivalent definition.

We first observe that, since $K > K_1 > 0$, if the minimum is attained for some $\boldsymbol{a} = \boldsymbol{a_1}$ then $(\boldsymbol{a_1} \cdot \nabla\phi(\boldsymbol{x_0})) < 0$. Thus,

$$\nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{a_1}}{K(\boldsymbol{x_0}, \boldsymbol{a_1})} \geq \nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{a_1}}{K_1}.$$

Let $\boldsymbol{b} = \frac{-\nabla\phi(\boldsymbol{x_0})}{\|\nabla\phi(\boldsymbol{x_0})\|}$. Since $\boldsymbol{a_1}$ is the minimizer, we have

$$\nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{a_1}}{K(\boldsymbol{x_0}, \boldsymbol{a_1})} \leq \nabla\phi(\boldsymbol{x_0}) \cdot \frac{\boldsymbol{b}}{K(\boldsymbol{x_0}, \boldsymbol{b})} \leq \frac{-\|\nabla\phi(\boldsymbol{x_0})\|}{K_2}.$$

Therefore,

$$\boldsymbol{a_1} \cdot \nabla\phi(\boldsymbol{x_0}) \leq -\|\nabla\phi(\boldsymbol{x_0})\|\frac{K_1}{K_2}.$$

Remark 2.2.23. We have just established a bound on the angle between the characteristic and the gradient line. If the gradient $\nabla u(\boldsymbol{x_0})$ exists then $\nabla u(\boldsymbol{x_0}) = \nabla \phi(\boldsymbol{x_0})$. Therefore,

$$\boldsymbol{a_1} \cdot \nabla u(\boldsymbol{x_0}) \leq -\|\nabla u(\boldsymbol{x_0})\| \frac{K_1}{K_2}.$$

If γ is the angle between $\nabla u(\boldsymbol{x_0})$ and $(-\boldsymbol{a_1})$ then

$$\cos(\gamma) \geq \frac{K_1}{K_2}.$$

This result is truly important for our presentation, since it will serve as a basis for building the single-pass methods introduced in chapters 4 and 6.

2.2.7 Numerical methods

The central issues of this thesis are the applicability, efficiency and convergence properties of various numerical methods for a wide class of static Hamilton-Jacobi PDEs (defined in section 1.4). As shown in the preceding sections, the Hamilton-Jacobi-Bellman PDE 2.24 belongs to that class[3]. The unique viscosity solution of that PDE is precisely the value function $u(\boldsymbol{x})$ for the corresponding optimal trajectory problem.
In this section we survey three first-order methods, which are typical in the context of such control-theoretic problems. We observe that

- **Dijkstra's method on the approximating grid** is single-pass, but, as the grid is refined, the obtained numerical solution $U(\boldsymbol{x})$ does not converge to $u(\boldsymbol{x})$;
- **Gonzales-Rofman method** yields a numerical solution $U(\boldsymbol{x})$ converging to $u(\boldsymbol{x})$, but is not single-pass; therefore, it is significantly more computationally expensive;
- **Tsitsiklis' method** is single-pass, efficiently produces numerical solutions for $U(\boldsymbol{x})$ converging to $u(\boldsymbol{x})$, but only for the isotropic running cost functions; therefore, it can be used for the Eikonal equation 2.18 only.

Our new methods, which are both single-pass and applicable to the general Hamilton-Jacobi-Bellman PDEs 2.15, will be introduced in chapters 4 and refchap:HybridMethod.

[3] In section 6.1 we will actually show that everything in that class of equations can be viewed as a Hamilton-Jacobi-Bellman PDE stemming from the optimal trajectory problem.

2.2.8 Why Dijkstra's method cannot be used here.

Not even in the Eikonal case (i.e., $K(\boldsymbol{x}, \boldsymbol{a}) = K(\boldsymbol{x})$), alas...
It seems so simple and so reasonable:

- lay out a uniform Cartesian grid X, with the grid points $\boldsymbol{x_{ij}} = (hi, hj)$;
- consider the grid points to be the nodes of the network;
- set the transition costs to be $K(\boldsymbol{x_{ij}}, \tilde{\boldsymbol{x}}) = hK(\boldsymbol{x_{ij}})$ if $\tilde{\boldsymbol{x}}$ is one of the four neighbors of $\boldsymbol{x_{ij}}$, and $K(\boldsymbol{x_{ij}}, \tilde{\boldsymbol{x}}) = \infty$ otherwise.
- set the termination cost to be $q(\boldsymbol{x})$ for all the nodes on $\partial\Omega$;
- use Dijkstra's method find the value function U for the resulting "shortest path on the network" problem.

It works, but with a little caveat: as the grid is refined, U does not converge to u. Figure 2.4 below shows the level sets for of U obtained by this version of Dijkstra's algorithm for the simplest possible case $K(\boldsymbol{x}, \boldsymbol{a}) = 1$, and the boundary condition $u(0, 0) = 0$. This picture does not improve as the grid is refined, and the reason for this is that only those trajectories running along network links are considered during the optimization. Thus, any trajectory which requires "diagonal" motion will not be modeled no matter how small the grid size h is[4].

As demonstrated in [46], these numerical approximations U can be formally shown to approximate the solution of the PDE

$$\max(|u_x(x, y)|, |u_y(x, y)|) = K(x, y),$$

which is obviously not equivalent to the Eikonal PDE

$$\|\nabla u(x, y)\| = K(x, y).$$

[4]Of course, as the grid is refined, we can layout a path on the grid, which is arbitrarily close to the optimal trajectory. However, the length of that path will not converge to the length of the true optimal trajectory even as $h \to 0$.

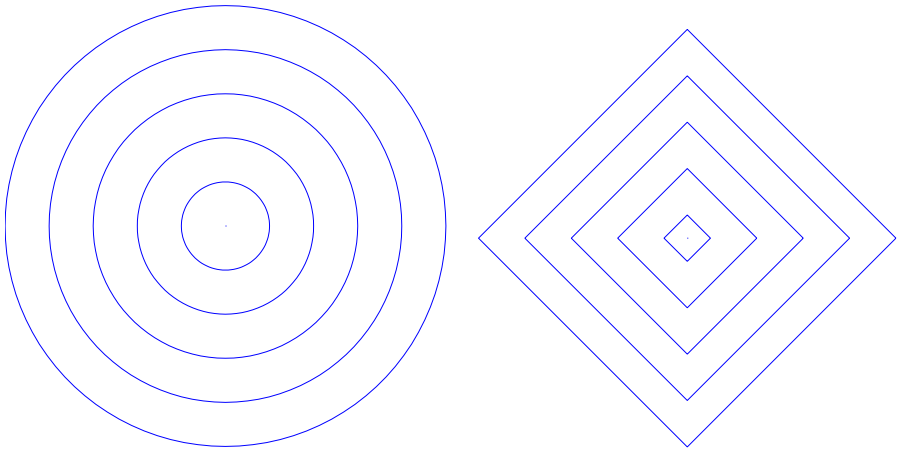


Figure 2.4: The "distance from the origin": Euclidean (left) and as found by Dijkstra's method (right). The latter is often referred to as a "Manhattan distance".

2.2.9 Gonzales-Rofman iterative method

Here we provide a derivation of the method based on a "naive" application of Bellman's optimality principle; the rigorous derivation and the proof of convergence can be found in [24][5].

We assume that a triangulated mesh X of the diameter h is defined on Ω. For every mesh point $\boldsymbol{x} \in X$ we define $S(\boldsymbol{x})$ to be a set of all the simplexes in the mesh adjacent to $\boldsymbol{x}$, (i.e., the simplexes, which have $\boldsymbol{x}$ as one of their vertices). If $s \in S(\boldsymbol{x})$, we will use the notation $\boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$ for the other vertices of the simplex s.

> If we do not change our direction, we are likely to end up where we are headed.
>
> *Chinese Proverb*

The key assumption in deriving this discretization is that, as the vehicle starts to move from a mesh point $\boldsymbol{x}$, the direction of motion $\boldsymbol{a}$ does not change until the vehicle reaches the edge of the simplex.

[5]The original Gonzales-Rofman iterative method was designed for a slightly different control-theoretic problem, in which the running cost is time-discounted, and the resulting Hamilton-Jacobi-Bellman PDE has the form $\lambda v(\boldsymbol{x}) = \min_{\boldsymbol{a} \in S_1}\{\nabla v(\boldsymbol{x}) \cdot \frac{\boldsymbol{a}}{K(\boldsymbol{x},\boldsymbol{a})}\} + 1$, where the value function $v(\boldsymbol{x})$ is related to the value function $u(\boldsymbol{x})$ of our version of the optimal trajectory problem by Kruzhkov's transform: $v(\boldsymbol{x}) = 1 - e^{-\lambda u(\boldsymbol{x})}$. We note that the need for this "technical trick" seems unavoidable, since even the existence of the viscosity solution to our HJB PDE 2.24 is normally demonstrated by employing Kruzhkov's transform and then proving the existence of the viscosity solution $v(\boldsymbol{x})$. See [21, 30] for details.

Suppose that $\boldsymbol{a}(t)$ is the optimal control for the point $\boldsymbol{x}$ and the corresponding optimal trajectory $\boldsymbol{y}'(t) = \boldsymbol{a}(t)$ intersects the edge $\boldsymbol{x_{s,1}x_{s,2}}$ of the simplex $s \in S(\boldsymbol{x})$ at some point $\tilde{\boldsymbol{x}} = \zeta \boldsymbol{x_{s,1}} + (1-\zeta)\boldsymbol{x_{s,2}}$ (see Figure 2.5).

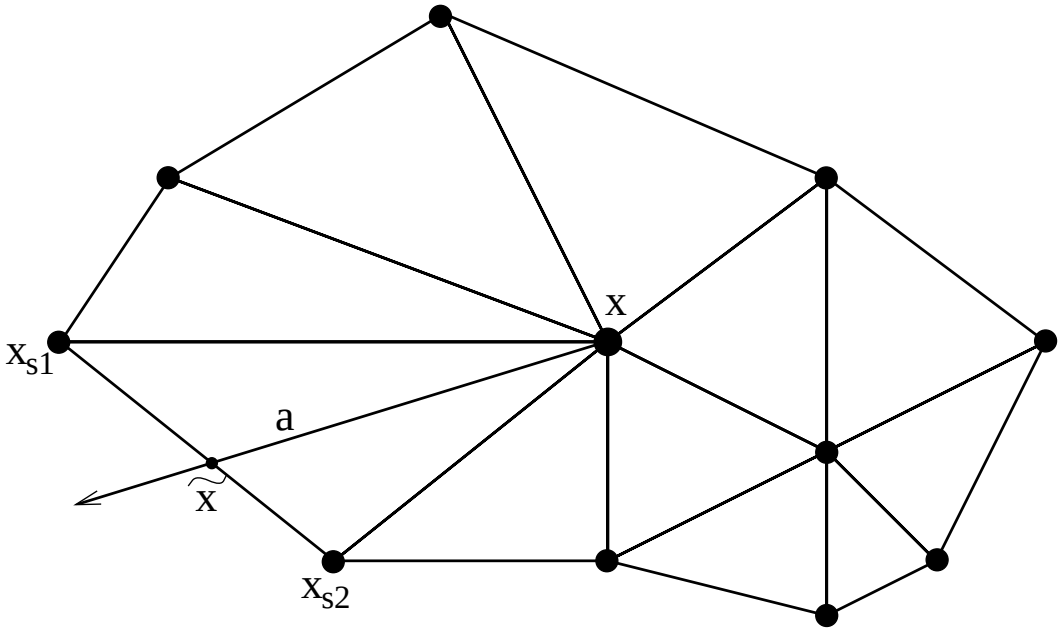


Figure 2.5: Gonzales-Rofman discretization: just keep going straight until you hit the edge of the simplex.

Define $\tau(\zeta) = \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| = \|(\zeta \boldsymbol{x_{s,1}} + (1-\zeta)\boldsymbol{x_{s,2}}) - \boldsymbol{x}\|$. If the vehicle is assumed to move locally along the straight lines, then $\boldsymbol{a}(t) = \boldsymbol{a}_\zeta = \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)}$. By the optimality of that trajectory,

$$u(\boldsymbol{x}) \approx \int_0^{\tau(\zeta)} K(\boldsymbol{y}(s), \boldsymbol{a}_\zeta) ds + u(\tilde{\boldsymbol{x}}) \approx \tau(\zeta) K(\boldsymbol{x}, \boldsymbol{a}_\zeta) + u(\tilde{\boldsymbol{x}}).$$

We note that $u(\tilde{\boldsymbol{x}})$ can be linearly approximated using $u(\boldsymbol{x_{s,1}})$ and $u(\boldsymbol{x_{s,2}})$. Invoking the optimality principle, we can now write the equation for the numerical approximation U:

$$\begin{aligned} U(\boldsymbol{x}) &= \min_{s \in S(\boldsymbol{x})} V_s(\boldsymbol{x}); \\ V_s(\boldsymbol{x}) &= \min_{\zeta \in [0,1]} \left\{ \tau(\zeta) K(\boldsymbol{x}, \boldsymbol{a}_\zeta) + \zeta U(\boldsymbol{x_{s,1}}) + (1-\zeta) U(\boldsymbol{x_{s,2}}) \right\}. \end{aligned} \tag{2.27}$$

Remark 2.2.24. As often is the case with the control-theoretic numerical methods (e.g., [4, 12, 22]), the above equation is obtained by applying the optimality criterion, rather than by discretizing the corresponding Hamilton-Jacobi-Bellman PDE. Nevertheless, any

approach, which yields the value function $u(x)$ in the limit, will automatically solve the PDE by the uniqueness of the viscosity solution [16, 15].

Remark 2.2.25. Even if the viscosity solution $u(\boldsymbol{x})$ is absolutely smooth, the above method is at most first order accurate since the first order approximations were used both for $u(\tilde{\boldsymbol{x}})$ and for the cost of using the control $\boldsymbol{a}(\cdot)$. Similar higher-order control-theoretic numerical methods can be found in [23].

The discretized equation 2.27 has to be satisfied at every mesh point in X; this leaves us with a coupled system of M non-linear equations, which usually have to be solved simultaneously through the iterations. Due to the structure of the system, each iteration involves solving a local minimization problem for each mesh point and even in the simplest problems the number of iterations will be proportional to the diameter of the mesh-graph. The number of iteration can be reduced using Gauss–Seidel relaxation (as in [24]), but we know of no theoretical guarantees of the rate of convergence.

2.2.10 Tsitsiklis' Algorithm for the isotropic case

For a special case of the Eikonal equation, the iterative approach is not the only option. Two fast methods are available for this problem. Sethian Fast Marching Method, based on the upwind finite difference schemes will be discussed in section 3.4.2. Here we present a single-pass version of the Gonzales-Rofman scheme for the isotropic running cost. This method is a modification of a fast algorithm developed by Tsitsiklis in [53] adapted for the unstructured meshes. Tsitsiklis has shown that the numerical solution U obtained by the Gonzales-Rofman scheme for the Eikonal equation has a property similar to the causality property (2.2.19) of the viscosity solution to that equation[6]:

Property 2.2.26 (Causality). *If $V_s(\boldsymbol{x})$ is computed by the formula 2.27, then $V_s(\boldsymbol{x}) \geq \max\{U(\boldsymbol{x_{s,1}}), U(\boldsymbol{x_{s,2}})\}$.*

This is not surprising, since the equation 2.27 enforces *upwinding* : the minimizing ζ corresponds to an approximate optimal trajectory (characteristic); thus, the numerical method computes $U(\boldsymbol{x})$ based on the simplex $s \in S(\boldsymbol{x})$ which contains the characteristic for $\boldsymbol{x}$. By the property 2.2.19, this automatically means that ∇U is pointing **from** the simplex s.

[6] The actual property proven in [53] is slightly different since Tsitsiklis' method was formulated on a uniform Cartesian grid.

Define $N_- = \{\boldsymbol{x_i} \mid \boldsymbol{x_i} \in N(\boldsymbol{x}) \text{ and } U(\boldsymbol{x_i}) < U(\boldsymbol{x})\}$ and $NU_- = \{U(\boldsymbol{x_i}) \mid \boldsymbol{x_i} \in N_-(\boldsymbol{x})\}$. Since $U(\boldsymbol{x}) \leq V_s(\boldsymbol{x})$, the property 2.2.26 allows to conclude that $U(\boldsymbol{x})$ depends only upon $NU_-(\boldsymbol{x})$.

The same construction used in section 2.1.3 to build the original Dijkstra's algorithm can be employed here to obtain the following single-pass method.

All the mesh points are divided into three classes: *Far* (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*), for which V has already been computed, but it is still unclear if $V = U$. For every *Considered* $\boldsymbol{x}$, we define the set $\mathrm{NS}(\boldsymbol{x}) = \{s \in S(\boldsymbol{x}) \mid \boldsymbol{x_{s,1}} \text{ and } \boldsymbol{x_{s,2}} \text{ are } \textit{Accepted}\}$. We also will use a set $S(\boldsymbol{x_1}, \boldsymbol{x_2})$ to denote the simplexes adjacent to both of these mesh points.

1. Start with all the mesh points in *Far*.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{s \in NS(\boldsymbol{x})} V_s(\boldsymbol{x}). \tag{2.28}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).
6. Move the *Far* mesh points adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$) into *Considered*.
7. Re-evaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$)

$$V(\boldsymbol{x}) := \min \left\{ V(\boldsymbol{x}), \min_{s \in \left(S(\boldsymbol{x}, \bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x})\right)} V_s(\boldsymbol{x}) \right\}. \tag{2.29}$$

8. If *Considered* is not empty then go to 4.

We note that the resulting algorithm

- produces the numerical solution U in $O(M \log(M))$ steps (same complexity analysis as for Dijkstra's method);

- produces the numerical solution U which converges to u as the diameter of the mesh tends to zero (follows from the convergence properties of the Gonzales-Rofman scheme);
- is only first order (Gonzales-Rofman scheme is based on the first order approximations; we are currently unaware of any other control-theoretic discretization, which would posses the property 2.2.26);
- is only applicable for the Eikonal equation (the impossibility to directly apply this method to anisotropic problems will be further discussed in section 4.1).

Chapter 3

Front Propagation Problems

3.1 Huygens' principle

Consider a simple curve Γ_t evolving in R^2. We will assume that the curve moves in the direction normal to itself with some speed $F(\boldsymbol{x}, \boldsymbol{n})$[1], where $\boldsymbol{n}$ is the "outwards pointing" unit vector normal to the curve as it passes through the point $\boldsymbol{x}$.

Such a definition is meaningful only for a smooth curve. Wherever Γ_t does not have a well-defined tangent a more elaborate construction (based on a variant of *Huygens' principle*) is required:

1. at every point $\boldsymbol{x} \in \Gamma_t$ where the vector $\boldsymbol{n}$ is well-defined we construct a circle of radius $|F(\boldsymbol{x}, \boldsymbol{n})|\Delta t$;
2. taking the envelope of all such circles, one obtains an approximation of $\Gamma_{t+\Delta t}$;
3. the limit of such approximations as $\Delta t \to 0$ provides a definition of the curve's evolution.

Remark 3.1.1. Note that Huygens' principle by itself does not keep track of whether the front is advancing or retreating: the envelope referred to in the above description should be a "signed envelope" since F might be changing sign (see the lower left picture in Figure 3.1, for example).

[1] In general, the function F might depend upon local geometric characterization of the curve (e.g., normal direction, curvature, etc), global properties of the curve (e.g., some integral computed along Γ_t), or the properties of the medium, in which the curve is propagating. In this discussion we will consider only F dependent upon the current position and orientation of Γ_t. A collection of methods for the more general front propagation problems can be found in [46].

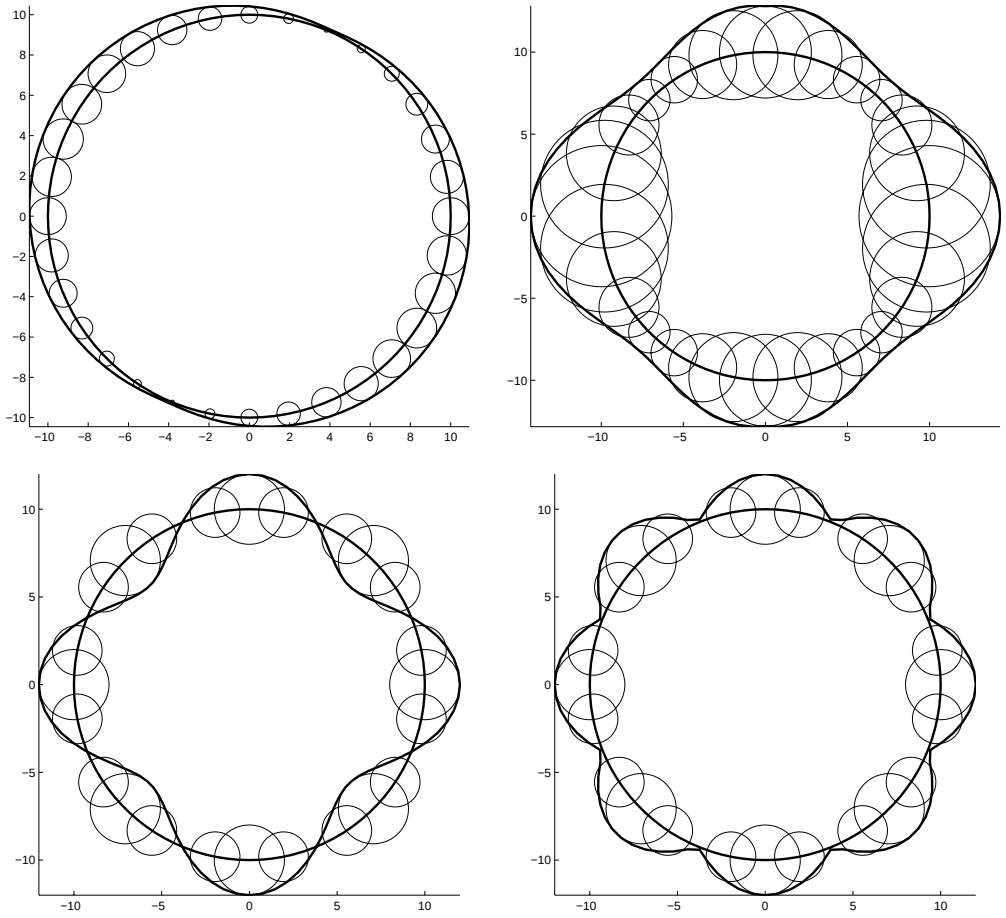


Figure 3.1: Huygens' construction: a circle evolving with different speeds in the normal direction.

3.2 The boundary value problem formulation

> Anyone informed that the universe is expanding and contracting in pulsations of eighty billion years has a right to ask, "What's in it for me?"
>
> PETER DE VRIES (1910 - 1993)
> US NOVELIST, EDITOR

An important subclass of the front propagation problems consists of all the applications, where the speed function F never changes sign. If the function F is strictly positive (or negative), then the front always expands (or contracts). An important implication is

that such a front can pass through each point only once[2]. Thus, we can define $u(\boldsymbol{x})$ to be the time when the interface passes through the point $\boldsymbol{x}$ and derive a partial differential equation[3] formally satisfied by u.

We first note that the level sets of u correspond to the positions of the front Γ at different times:

$$u(\boldsymbol{x}) = t \iff \boldsymbol{x} \in \Gamma_t.$$

Therefore, if F is always nonnegative, the outwards unit normal vector can be expressed as

$$\boldsymbol{n}(\boldsymbol{x}) = \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|} \tag{3.1}$$

For some small τ, consider a point $\bar{\boldsymbol{x}} = \boldsymbol{x} + \tau \boldsymbol{n}(\boldsymbol{x})$. Since the front is advancing in the normal direction with the speed $F(\boldsymbol{x}, \boldsymbol{n})$, we know that $u(\bar{\boldsymbol{x}}) \approx u(\boldsymbol{x}) + \frac{\tau}{F(\boldsymbol{x}, \boldsymbol{n})}$. On the other hand, since $\boldsymbol{n}$ is a unit vector in the direction of ∇u, we know that $\|\nabla u\| \approx \frac{u(\bar{\boldsymbol{x}}) - u(\boldsymbol{x})}{\tau}$. Thus, passing to the limit as $\tau \to 0$, we obtain the PDE formally satisfied by u:

$$\begin{gathered} \|\nabla u(\boldsymbol{x})\| F\left(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}\right) = 1, \\ u = 0 \text{ on } \Gamma_0. \end{gathered} \tag{3.2}$$

The above equation is in fact a static Hamilton-Jacobi PDE of the form $H(\nabla u, \boldsymbol{x}) = 1$, where the Hamiltonian H is homogeneous of degree one in the first argument. To interpret Eqn 3.2 where ∇u does not exist one normally uses the unique viscosity solution, as defined in [15, 16]. As follows from the results in [20], the viscosity solution u of Eqn 3.2 will correspond to the evolution of Γ_0 defined by Huygens' principle. In general, the Hamiltonian $H(\nabla u, \boldsymbol{x}) = \|\nabla u\| F\left(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|}\right)$ is not convex. As shown in [20], such a Hamiltonian can be always considered as a result of a differential game model. To the best of our knowledge, there are currently no single-pass numerical methods for these models although some iterative schemes are well known (see [5], for example). However, if the speed function F is such that the Hamiltonian is convex and for some constants F_1, F_2

$$0 < F_1 < F(\boldsymbol{x}, \boldsymbol{p}) < F_2 < \infty \qquad\qquad \forall \boldsymbol{x}, \boldsymbol{p}$$

then Eqn 3.2 belongs to the class of PDEs described in section 1.4. In Chapter 6 we will show that such an equation can always be considered as a Hamilton-Jacobi-Bellman PDE

[2] The *entropy criterion* formulated in [40] for the flame propagation in combustion problems is a perfect example of this principle: "Once a particle burns, it remains burnt."

[3] Our derivation is close to the one produced in [46].

for a suitable optimal trajectory problem. Therefore, this subclass of front propagation problems can be treated in the framework of control-theory, and the single-pass methods defined in chapters 4 and 6 are applicable.

3.3 The glory of isotropy: Eikonal revisited.

Suppose that the front propagation is *isotropic*, i.e., the speed of the front motion in the normal direction is independent of the orientation of the front:

$$F(\boldsymbol{x}, \boldsymbol{n}) = F(\boldsymbol{x}).$$

In this case, the static Hamilton-Jacobi PDE 3.2 immediately reduces to the Eikonal equation:

$$\|\nabla u(\boldsymbol{x})\| F(\boldsymbol{x}) = 1. \tag{3.3}$$

Setting $K(\boldsymbol{x}) = \frac{1}{F(\boldsymbol{x})}$, we see the connection to the Eikonal equation obtained from the control-theoretic point of view in section 2.2.4. We note that

- $F_1 = \frac{1}{K_2}$, $F_2 = \frac{1}{K_1}$. Therefore, the anisotropy coefficient $\Upsilon = \frac{K_2}{K_1} = \frac{F_2}{F_1}$.
- If we consider the "constant cost, variable speed" optimal trajectory problem (from section 2.2.5), then $F(\boldsymbol{x}) = f(\boldsymbol{x})$, $F_1 = f_1$, $F_2 = f_2$. We will see in Chapter 6 that the relationship is much more complicated in the anisotropic case.
- Not surprisingly, the case $F(\boldsymbol{x}) = 1$, which describes the motion of the front with the unit speed, can be viewed as a problem of finding the minimum distance $d(\boldsymbol{x})$ from $\boldsymbol{x}$ to Γ_0.

3.4 Numerical methods for isotropic front propagation.

This connection between the isotropic front propagation and isotropic min-time optimal trajectory problems is quite well known. For example, in [21] it is suggested to use the iterative control-theoretic methods to obtain the numerical solution for the isotropic flame propagation models. Of course, Tsitsiklis' algorithm (1995) can also be used in this setting to produce the first-order approximation efficiently. However, some of the more common front propagation problems come from combustion, fluid dynamics and crystal

growth. Finite-difference schemes, which primarily approximate the PDE rather than some optimality principle, are historically much more common in these application domains. In this section we will consider a number of such methods, including several versions of the Fast Marching Method - a single-pass technique for solving the Eikonal and related problems introduced by Sethian (1996). The higher-order upwinding finite-difference update formulas used in this method will be required for building the hybrid methods (Chapter 6), which rely on both the control-theoretic and the front propagation perspectives.

Honest differences are often a healthy sign of progress.

MAHATMA GANDHI (1869 - 1948)
INDIAN PHILOSOPHER

3.4.1 Upwinding finite difference discretization.

On a uniform Cartesian grid X of grid size h, the Eikonal equation 3.3 can be approximated as follows:

$$max(D^{-x}U_{i,j}, -D^{+x}U_{i,j}, 0)^2 + max(D^{-y}U_{i,j}, -D^{+y}U_{i,j}, 0)^2 = F_{i,j}^{-2}, \tag{3.4}$$

where $D^{-x}U_{i,j} = \frac{U_{i,j}-U_{i-1,j}}{h}, D^{+x}U_{i,j} = \frac{U_{i+1,j}-U_{i,j}}{h}, D^{-y}U_{i,j} = \frac{U_{i,j}-U_{i,j-1}}{h}, D^{+y}U_{i,j} = \frac{U_{i,j+1}-U_{i,j}}{h}$.

This numerical first-order scheme, given in [38], is based on a Godunov-type monotone upwinding scheme[4]. The convergence of such monotone, upwinding schemes to the viscosity solution can be shown based on the results in [6].

Remark 3.4.1. Note that, in contrast with the control-theoretic methods, this scheme is derived by directly approximating the components of ∇u with the finite difference operators. The choice of *upwinding finite difference operators* is natural if one considers the connection of Hamilton-Jacobi PDEs to the hyperbolic conservation laws.

Since the equation 3.4 has to hold at every grid point, it presents a challenge of solving a coupled system of M non-linear equations. In [38] it was proposed to solve this system iteratively. Sethian's original Fast Marching Method [43] is based on an observation

[4]Throughout this work, the term *upwinding* is used to signify that the value of $U_{i,j}$ is computed using the simplex, which contains the characteristic for the point $\boldsymbol{x_{i,j}}$. It is easy to see that the above formula computes $U_{i,j}$ using the simplex from which the gradient $\nabla u(\boldsymbol{x_{i,j}})$ is pointing. Since, for the isotropic case, the gradient direction and the characteristic directions are exactly the opposite, this is equivalent to demonstrating that this scheme is upwinding.

that, as a result of the upwinding, the system specified by Eqn 3.4 can be effectively decoupled and solved much more efficiently.

3.4.2 Sethian's Fast Marching Method.

As it was proven in [43], the value $U_{i,j}$ satisfying the equation 3.4 can be computed "quadrant by quadrant", as follows:
Define $S(\boldsymbol{x_{i,j}})$ to be the set of four right triangles of the form $\boldsymbol{x_{i\pm1,j}x_{i,j}x_{i,j\pm1}}$. Define the upwinding update V_s from the triangle $s \in S(\boldsymbol{x_{i,j}})$ to be the solution of the equation

$$\left(\max\left(\frac{V_s - U(\boldsymbol{x_{s1}})}{h}, 0\right)\right)^2 + \left(\max\left(\frac{V_s - U(\boldsymbol{x_{s2}})}{h}, 0\right)\right)^2 = F^{-2}(\boldsymbol{x_{i,j}}).$$

Define

$$U_{i,j} = \min_{s \in S(\boldsymbol{x_{i,j}})} V_s.$$

Moreover, the following key observation in [43] shows that the numerical solution mirrors the property 2.2.19 of the Eikonal equation.

Observation 3.4.2 (Causality). The upwinding nature of the finite difference operators employed in (3.4) ensures that $U_{i,j}$ depends only on those values in $NU(\boldsymbol{x_{i,j}})$, which are smaller than $U_{i,j}$.

For example, if $U_{i,j+1} \geq U_{i,j}$ then $D^{+y}U_{i,j} \geq 0$ and $\max(D^{-y}U_{i,j}, -D^{+y}U_{i,j}, 0) = \max(D^{-y}U_{i,j}, 0)$.

Again, we are in the situation, where the system could be solved in $O(M)$ operations if only the ordering of $\boldsymbol{x_{i,j}}$ based on $U_{i,j}$ were known. Of course, the ordering is not a priori known; hence Fast Marching Method determines it "one grid point at a time" in the spirit of Dijkstra's method, as described in section 2.1.3.

All the grid points are divided into three classes: *Far* (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*), for which V has already been computed, but it is still unclear if $V = U$. For every *Considered* $\boldsymbol{x}$, we define the set $\mathrm{NS}(\boldsymbol{x}) = \{s \in S(\boldsymbol{x}) \mid \boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$ are *Accepted*$\}$. If $\boldsymbol{x_1}$ is adjacent to $\boldsymbol{x_2}$, we will use $S(\boldsymbol{x_1}, \boldsymbol{x_2})$ to denote the two "quadrant triangles" for $\boldsymbol{x_1}$, which have $\boldsymbol{x_2}$ as one of their vertices.

1. Start with all the grid points in *Far*.

2. Move the grid points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).

3. Move all the grid points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{s \in NS(\boldsymbol{x})} V_s(\boldsymbol{x}). \tag{3.5}$$

4. Find the grid point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.

5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).

6. Move the *Far* grid points adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$) into *Considered*.

7. Re-evaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$)

$$V(\boldsymbol{x}) := \min\left\{ V(\boldsymbol{x}), \min_{s \in \left(S(\boldsymbol{x}, \bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x})\right)} V_s(\boldsymbol{x}) \right\}; \tag{3.6}$$

or, if $S(\boldsymbol{x}, \bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x}) = \emptyset$,

$$V(\boldsymbol{x}) := \min\left\{ V(\boldsymbol{x}), U(\bar{\boldsymbol{x}}) + \frac{h}{F(\boldsymbol{x})} \right\}; \tag{3.7}$$

8. If *Considered* is not empty then go to 4.

This is the Fast Marching Method as described in [43].
We note that the resulting algorithm

- produces the numerical solution U in $O(M \log(M))$ steps (same complexity analysis as for Dijkstra's method);
- produces the numerical solution U which converges to u as the grid size h tends to zero (follows from the convergence properties of the discretization 3.4);
- is only first order accurate (the higher order extensions of the Fast Marching Method will be discussed in section 3.4.4);
- is formulated on a uniform Cartesian grid (the versions of the Fast Marching Method for the triangulated meshes will be described in section 3.4.3);
- is only applicable for the Eikonal equation (the impossibility to directly apply this method to general anisotropic problems will be demonstrated in section 4.1).

3.4.3 Fast Marching Method on triangulated meshes

The first extensions of Fast Marching Method to the triangulated meshes were performed in [27] for the meshes in R^2 and on two dimensional manifolds. A general version for the unstructured meshes in R^n was presented in [48]; the structure of our exposition is similar to the latter even though all the examples and formulas are derived for R^2.

General Strategy

In order to build the Fast Marching Method, we will first need to find a consistent upwinding approximation of the PDE on the unstructured mesh X. If $S(\boldsymbol{x})$ is the set of simplexes which have the mesh point $\boldsymbol{x}$ as one of their vertices, then we will look for a discretization in the form

$$U(\boldsymbol{x}) = \min_{s \in S(\boldsymbol{x})} V_s, \tag{3.8}$$

where V_s is the value of $U(\boldsymbol{x})$ computed using the values of $U(\boldsymbol{x_{s,1}})$ and $U(\boldsymbol{x_{s,2}})$, subject to the assumption that the characteristic for the mesh point $\boldsymbol{x}$ lies inside the simplex s. Thus, we

- show how to produce such V_s for a given simplex;
- derive an *upwinding criterion* (i.e., a way to verify that the value of V_s is consistent with the assumption about the characteristic direction);
- prove that for the acute meshes the *causality* is the result of upwinding (i.e., $U(\boldsymbol{x})$ depends **only** on the values in $NU(\boldsymbol{x})$ smaller than $U(\boldsymbol{x})$);
- proceed to build a single-pass algorithm for computing $U(\boldsymbol{x})$ on acute triangulated meshes;
- show how the causality can be restored for the non-acute simplexes.

Directional derivative approximations

The finite difference methods for Hamilton-Jacobi PDEs work by employing suitable upwinding approximations for the gradient. Therefore, our first concern is to find such an approximation on an unstructured mesh. Since there generally is no natural choice of the coordinate system for an unstructured mesh, we compute the gradient as a linear

combination of the directional derivatives. For any mesh point $\boldsymbol{x}$ and simplex $s \in S(\boldsymbol{x})$, the difference approximation of the directional derivative is obviously available for the directions $(\boldsymbol{x} - \boldsymbol{x_{s,1}})$ and $(\boldsymbol{x} - \boldsymbol{x_{s,2}})$. We note that the unit vectors $\boldsymbol{P_1} = \frac{\boldsymbol{x}-\boldsymbol{x_{s,1}}}{\|\boldsymbol{x}-\boldsymbol{x_{s,1}}\|}$ and $\boldsymbol{P_2} = \frac{\boldsymbol{x}-\boldsymbol{x_{s,2}}}{\|\boldsymbol{x}-\boldsymbol{x_{s,2}}\|}$ will be linearly independent. Consider the 2×2 nonsingular matrix P having $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ as its rows. Let $v_r(\boldsymbol{x})$ be the value of the directional derivative for the direction $\boldsymbol{P_r}$ evaluated at the point $\boldsymbol{x}$. Assuming that the function u is differentiable at $\boldsymbol{x}$, we have $P\nabla u(\boldsymbol{x}) = \boldsymbol{v}(\boldsymbol{x})$, for the vector $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$. Combining this with the Eikonal equation we can now write an equation for $\boldsymbol{v}(\boldsymbol{x})$:

$$\boldsymbol{v}(\boldsymbol{x})^T (PP^T)^{-1} \boldsymbol{v}(\boldsymbol{x}) = F^{-2}(\boldsymbol{x}). \tag{3.9}$$

The particular difference operators used to approximate v_r will depend upon the type of the mesh and will determine the order of convergence of the numerical method. We note, however, that if those difference operators depend upon $u(\boldsymbol{x})$ linearly, then the resulting discretized version of the Eikonal equation will be quadratic in terms of $U(\boldsymbol{x})$[5].

To obtain the discretized equation, we now replace each v_r with the corresponding difference approximation: $v_r \approx a_r U(\boldsymbol{x}) + b_r$, where b_r's linearly depend on the values of U (and possibly of ∇U for higher order schemes) at the mesh points $\boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$. For convenience let $Q = (PP^T)^{-1}$, $\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$, and $\boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$. If V_s is the approximation for $u(\boldsymbol{x})$ computed from the simplex s, then $\boldsymbol{v} \approx V_s \boldsymbol{a} + \boldsymbol{b}$ and the discretized version of Eqn 3.9 is the quadratic equation:

$$(\boldsymbol{a}^T Q \boldsymbol{a}) \left(V_s(\boldsymbol{x})\right)^2 + (2\boldsymbol{a}^T Q \boldsymbol{b}) V_s(\boldsymbol{x}) + (\boldsymbol{b}^T Q \boldsymbol{b}) = F^{-2}. \tag{3.10}$$

Upwinding Criteria

Recall that $U(\boldsymbol{x}) = V_s(\boldsymbol{x})$ if the characteristic for the mesh point $\boldsymbol{x}$ lies within the simplex s. The equation 3.10 for computing $V_s(\boldsymbol{x})$ based on $U(\boldsymbol{x_{s,1}})$ and $U(\boldsymbol{x_{s,2}})$ was also derived under that same assumption about the characteristic direction. It is, therefore, logical to request that the computed value of $V_s(\boldsymbol{x})$ should be consistent with the *upwinding criterion* : the computed approximate characteristic direction should lie within the simplex, which produced the approximation. Recall that for the Eikonal equation the characteristic

[5]This statement is also true for any unstructured meshes in R^n.

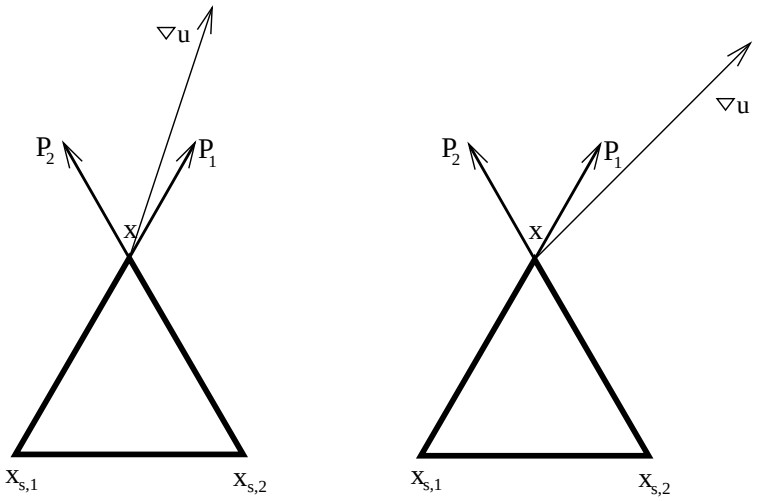


Figure 3.2: Examples of acceptable (left) and unacceptable (right) approximations for $\nabla u(\boldsymbol{x})$. In the latter case the upwinding requirement is not satisfied, and $U(\boldsymbol{x})$ will be computed using some other simplex.

direction is the opposite of the gradient direction, wherever ∇u is defined. Therefore, for the discretized Eikonal equation, the upwinding criterion can be strictly defined as follows: the computed value of $V_s(\boldsymbol{x})$ is acceptable if the update is coming from within the simplex s, i.e., only if the computed approximate $(-\nabla U(\boldsymbol{x}))$ lies inside the simplex. (See Figure 3.2.) This is equivalent to requiring that

$$\text{all of the components of the vector } Q\boldsymbol{v} \approx Q(V_s(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b}) \text{ are non-negative.} \tag{3.11}$$

In R^2 the above upwinding condition is equivalent to two inequalities: $(V_s(\boldsymbol{x})a_1 + b_1) \geq (\boldsymbol{P_1} \cdot \boldsymbol{P_2}^T)(V_s(\boldsymbol{x})a_2 + b_2)$ and $(V_s(\boldsymbol{x})a_2 + b_2) \geq (\boldsymbol{P_1} \cdot \boldsymbol{P_2}^T)(V_s(\boldsymbol{x})a_1 + b_1)$.

We now assume that the simplex s is acute. Hence, $(\boldsymbol{P_1} \cdot \boldsymbol{P_2}^T) > 0$ and, if both of the above inequalities are satisfied, it follows that $v_r \approx (V_s(\boldsymbol{x})a_r + b_r) \geq 0$, for $r = 1, 2$. Since v_r is the approximation for the directional derivative in the direction $\boldsymbol{P_r} = \frac{\boldsymbol{x} - \boldsymbol{x_{s,r}}}{\|\boldsymbol{x} - \boldsymbol{x_{s,r}}\|}$, we conclude that $V_s \geq \max(U(\boldsymbol{x_{s,1}}), U(\boldsymbol{x_{s,2}}))$, provided the upwinding condition is satisfied.

Thus, we have demonstrated that the discretized schemes of the form 3.10 possess the following *causality* property similar to the property 2.2.19 of the Eikonal equation.

Property 3.4.3 (Causality). *If V_s is computed from the acute simplex $s \in S(\boldsymbol{x})$ using the formula 3.10 and if the upwinding condition 3.11 is satisfied, then $V_s \geq \max(U(\boldsymbol{x_{s,1}}), U(\boldsymbol{x_{s,2}}))$.*

Since $U(\boldsymbol{x}) = \min_{s \in S(\boldsymbol{x})} V_s$, this means that $U(\boldsymbol{x})$ also depends only on the values

in $NU(\boldsymbol{x})$, which are smaller than $U(\boldsymbol{x})$. If the ordering of the mesh points based on the values of U were known in advance, the system 3.8 would be effectively de-coupled, and the equations could be solved one-by-one. However, since the ordering is not a priori known, we design the Fast Marching Method to determine it "one mesh point at a time", using the *causality* property. The algorithm structure is quite similar to the Fast Marching Method defined on the Cartesian Grids in the previous section. We include it here for the sake of completeness.

All the mesh points are divided into three classes: Far (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*), for which V has already been computed, but it is still unclear if $V = U$. For every *Considered* $\boldsymbol{x}$ we define the set $\mathrm{NS}(\boldsymbol{x}) = \{s \in S(\boldsymbol{x}) \mid \boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$ are *Accepted*$\}$. If $\boldsymbol{x_1}$ is adjacent to $\boldsymbol{x_2}$, we will use the notation $S(\boldsymbol{x_1}, \boldsymbol{x_2})$ to denote the set $S(\boldsymbol{x_1}) \bigcap S(\boldsymbol{x_2})$.

1. Start with all the mesh points in Far.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{s \in NS(\boldsymbol{x})} V_s(\boldsymbol{x}). \tag{3.12}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).
6. Move the Far mesh points adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$) into *Considered*.
7. Re-evaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$ (i.e., $\boldsymbol{x} \in N(\bar{\boldsymbol{x}})$)

$$V(\boldsymbol{x}) := \min\left\{ V(\boldsymbol{x}), \min_{s \in \left(S(\boldsymbol{x}, \bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x})\right)} V_s(\boldsymbol{x}) \right\}, \tag{3.13}$$

(where $V_s(\boldsymbol{x})$ is a solution of the equation 3.10 satisfying the upwinding condition 3.11)

or, if $S(\boldsymbol{x}, \bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x}) = \emptyset$,

$$V(\boldsymbol{x}) := \min\left\{ V(\boldsymbol{x}), U(\bar{\boldsymbol{x}}) + \frac{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|}{F(\boldsymbol{x})} \right\}; \tag{3.14}$$

8. If *Considered* is not empty then go to 4.

Extension to Obtuse Meshes

In all of the fast methods presented so far the de-coupling was made possible because the discretized upwind scheme possessed some property equivalent to the causality property (2.2.19) of the Eikonal equation. However, if the triangulated mesh contains simplexes with obtuse angles, then the causality relationship might not hold even in the limit (i.e., not even for the viscosity solution $u(\boldsymbol{x})$; recall that the property 2.2.19 required the simplex to be acute). Indeed, when the triangle $s = \boldsymbol{x}\boldsymbol{x_1}\boldsymbol{x_2}$ is obtuse, it is possible that $u(\boldsymbol{x}) < \max(u(\boldsymbol{x_1}), u(\boldsymbol{x_2}))$ even if the characteristic for the point $\boldsymbol{x}$ lies inside that triangle.

Consider, for example, the front advancing with unit speed in the direction $(1, 1)$ (suppose we are solving $\|\nabla u\| = 1$ in R^2 with the boundary condition $u = 0$ on the line $y = -x$). Consider the simplex $\boldsymbol{x}\boldsymbol{x_1}\boldsymbol{x_2}$ from Figure 3.3. The vector ∇u points **from** that simplex. Since the optimal direction for the Eikonal problems is $(-\nabla u)$, we see that the characteristic for the point $\boldsymbol{x}$ lies inside the triangle. At the same time, it is obvious from the picture that $u(\boldsymbol{x_1}) < u(\boldsymbol{x}) < u(\boldsymbol{x_2})$. If the discretization is upwinding, we need to compute $U(\boldsymbol{x})$ based on $U(\boldsymbol{x_1})$ and $U(\boldsymbol{x_2})$. However, if the mesh points are *Accepted* in the ascending order, this means that $U(\boldsymbol{x_2})$ will not be known in time to compute $U(\boldsymbol{x})$.

One possible solution is to build locally numerical support at obtuse angles, as was suggested in [27]. For an obtuse angle $\boldsymbol{x_1}\boldsymbol{x}\boldsymbol{x_2}$ (see Fig. 3.3a), consider its splitting section – an angle such that any ray inside it will split $\boldsymbol{x_1}\boldsymbol{x}\boldsymbol{x_2}$ into two acute angles. Find the closest *Accepted* mesh point in the splitting section and then use that point as if it were adjacent to $\boldsymbol{x}$. Thus, we would use simplex $\boldsymbol{x_1}\boldsymbol{x}\boldsymbol{x_6}$ in Fig. 3.3a.

There are some disadvantages to this method. First of all, the implementation for higher dimensions is rather cumbersome. Second, implementing it for triangulated surfaces requires an additional step of "unfolding" [27]. Third, the method is no longer confined to considering the mesh points immediately adjacent to $\boldsymbol{x}$ since we need to look back for an *Accepted* point in the splitting section. The upper bound for how far back we need to look in the splitting section is available but depends on the maximum angle of the triangulation – the wider angle corresponds to the narrower splitting section which is less likely to contain a mesh point near $\boldsymbol{x}$.

We observe that this splitting section method can be further improved by noting

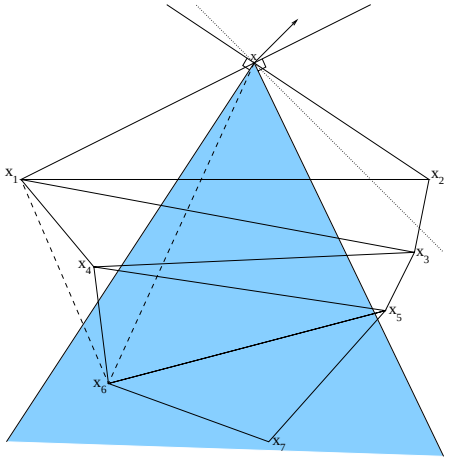


Fig.3.3a: Constructing splitting section for an obtuse angle

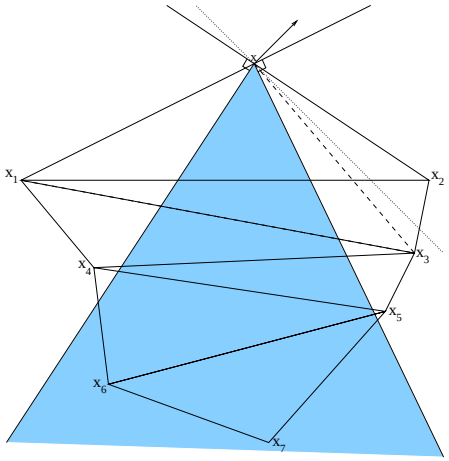


Fig.3.3b: Modified construction of splitting section

Figure 3.3: Splitting strategies

that the cause of the problem is not just the obtuse angle in the *defining simplex*, but the fact that some of the vertices of that simplex are not yet *Accepted*. Thus, it is not necessary to find an *Accepted* node in the splitting section; it is enough to find an *Accepted* node such that the resulting virtual simplex intersects the splitting section. This often allows to look back much less; thus, in Fig. 3.3b, for example, the mesh point $\boldsymbol{x_3}$ is the first *Accepted* point found in the splitting section, and hence the virtual simplex $\boldsymbol{x_1 x x_3}$ will be used to compute the update for $U(\boldsymbol{x})$.

We note that our construction works equally well on manifolds. In section 8.1.2 we present an example of computation, where the offsets equidistant from the bounding

box are found on a manifold, which represents a complex machine part. This computation illustrates the splitting section constructions; the triangulation is obtained by mapping a regular triangular mesh in the $x - y$ plane onto the surface, creating a large number of obtuse and near-degenerate triangles, including some with angles bigger than 160^o.

3.4.4 Higher order versions of the Fast Marching Method

We now create higher accuracy Fast Marching Methods by using higher order difference approximations for the directional derivatives. It would seem that such approximations can be used only if the solution u is sufficiently smooth; nonetheless, the fact that at some points ∇u is undefined does not prevent us from using this approach: u is differentiable almost everywhere, and characteristics never emanate from the shocks, i.e., no information is created at the shock. However, the order of the difference approximation does not always correspond to the order of convergence of the method. Still, such methods converge faster than the ones using the first order approximations [45]. We further discuss the rate of convergence of the higher accuracy methods in section 8.1.1. Our exposition repeats the results presented in [48].

Higher Order Methods on Cartesian Grid

A higher order Fast Marching Method on Cartesian grid was first presented by Sethian in [45]. Here we show that such methods can also be obtained from the directional derivative approximation perspective. For a Cartesian grid, the natural choice of the coordinate system will be aligned with the grid lines. Then, for any grid point $\boldsymbol{x}$, a direction vector $\boldsymbol{P_r}$ is always equal (up to the sign) to one of the canonical basis vectors. Thus, for $\boldsymbol{x}$ inside the grid (i.e., away from the boundary) both points $\boldsymbol{x_{r,1}} = \boldsymbol{x} - h\boldsymbol{P_r}$ and $\boldsymbol{x_{r,2}} = \boldsymbol{x} - 2h\boldsymbol{P_r}$ are also present in the grid. Then we can use the well-known second order difference approximation for the directional derivative

$$v_r(\boldsymbol{x}) \approx \frac{3u(\boldsymbol{x}) - 4u(\boldsymbol{x_{r,1}}) + u(\boldsymbol{x_{r,2}})}{2h}. \tag{3.15}$$

Using the notation introduced in section 3.4.3, we can write

$$a_r = \frac{3}{2h}; \qquad b_r = \frac{-4u(\boldsymbol{x_{r,1}}) + u(\boldsymbol{x_{r,2}})}{2h}.$$

Since this approximation is valid only inside the domain we need to have the exact values of u for the grid points near the boundary in order to start the algorithm. If these values

are not available, we can use the first order Fast Marching Algorithm with much smaller mesh size to obtain the second order accurate approximations of u at those points.

We note that the same higher order difference approximations can be used even for non-Cartesian grids provided all the grid points lie on the straight lines and are equidistant on those lines.

Higher Order Methods on Unstructured Mesh

Typically, we do not have orthogonal difference operators on an unstructured mesh. Fortunately, we can still build higher order directional derivative approximations by using the gradient information at the grid points adjacent to $\boldsymbol{x}$.

Consider a grid point $\boldsymbol{x_r}$ adjacent to $\boldsymbol{x}$ and the corresponding directional vector $\boldsymbol{P_r} = \frac{\boldsymbol{x}-\boldsymbol{x_r}}{\|\boldsymbol{x}-\boldsymbol{x_r}\|}$. Supposing that both $u(\boldsymbol{x_r})$ and $\nabla u(\boldsymbol{x_r})$ are known, we can write a second order approximation for the directional derivative

$$v_r(\boldsymbol{x}) \approx 2\frac{u(\boldsymbol{x}) - u(\boldsymbol{x_r})}{\|\boldsymbol{x} - \boldsymbol{x_r}\|} - \boldsymbol{P_r} \cdot \nabla u(\boldsymbol{x_r}). \tag{3.16}$$

Using the notation introduced in section 3.4.3, we can write

$$a_r = \frac{2}{\|\boldsymbol{x} - \boldsymbol{x_r}\|}; \qquad b_r = \frac{-2u(\boldsymbol{x_r})}{\|\boldsymbol{x} - \boldsymbol{x_r}\|} - \boldsymbol{P_r} \cdot \nabla u(\boldsymbol{x_r}).$$

We can also compute the second order accurate approximation for the gradient at $\boldsymbol{x}$, namely $\nabla u(\boldsymbol{x}) = P^{-1}\boldsymbol{v} \approx P^{-1}(u(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b})$, provided u($\boldsymbol{x}$) is known with at least the second order accuracy as well. Thus, as the algorithm runs, we will store for each *Accepted* grid point the computed values of both u and ∇u to be later used when recomputing the *Considered* points.

Since this approach requires information about the gradient we need to have the exact values of ∇u for the grid points on the boundary in order to start the algorithm. If these values are not available, we can use the first order Fast Marching Algorithm with much denser mesh in the narrow band near the boundary to obtain the accurate approximations of ∇u at the points in that narrow band.

Finally, we note that an additional step of "gradient mapping" is required to use this higher order method on non-smooth triangulated surfaces [48].

Remark 3.4.4. Since the above approximations are second order accurate only away from a singularity, we note that the exact (or second order accurate) values of u are also needed

for the grid points in the narrow band around each singularity at the boundary. The width of the narrow band should stay constant as the mesh is refined to observe the second order convergence to the viscosity solution.
This requirement equally applies to the higher order methods on Cartesian grids and on unstructured meshes. On unstructured meshes, the second order accurate values of ∇u are also required in the same narrow band.

3.4.5 Comparing two fast methods for the Eikonal equation.

So far we have considered two different single-pass methods for the Eikonal equation: Tsitsiklis' algorithm and Sethian's Fast Marching Method. Each of these methods is effectively de-coupling the system of equations, which are the discrete analogue of the Eikonal PDE. Thus, it is not surprising that there are several similarities in their structure and properties:

- Each method is based on an observation that the particular discretization possesses a *causality* property similar to that of the Eikonal equation.
- Both methods compute the numerical solutions converging to the viscosity solution.
- Both use heap-sort data structures to achieve Dijkstra-like efficiency.

We also summarize here the main differences between these two single-pass numerical methods:

- The discretizations are different and so are the underlying formulations (front propagation vs. optimal trajectories).
- An additional upwinding criterion is necessary to make Fast Marching Methods work; no such criterion is necessary for Tsitsiklis' algorithm. The reason for this is that the control-theoretic discretization 2.27 works by directly approximating the optimal trajectory (characteristic) rather than the gradient.
- In many applications, tracking the front propagation is only a part of a bigger problem, and upwind finite difference numerical methods are used for subsequent calculations. These finite differences can be saved for the later usage if the Fast Marching Method is used to solve the Eikonal equation.

- The quadratic update formula is readily available for the Fast Marching Method on an arbitrary mesh in R^n; for Tsitsiklis' algorithm, the corresponding update formula has to be produced by solving the local $(n-1)$-dimensional quadratic optimization problem. We note that, on a uniform Cartesian grid in R^2, the quadratic update formula is the same for both methods. This is not the case for unstructured meshes.
- Many extensions of the Fast Marching Method are available, including implementations on unstructured meshes, in R^n and on manifolds[6].
- It is much easier to build higher-order versions of the Fast Marching Method, since numerous upwind finite difference operators are available.

Our new hybrid methods for solving the more general anisotropic problems (Chapter 6) will use the upwind finite difference discretizations (similar to those employed in the Fast Marching Method), while determining the range of possible characteristic directions, based on the control-theoretic considerations.

3.5 Simple anisotropic example: Eikonal on a manifold.

We have demonstrated that the Fast Marching Method can be implemented not only on regular grids, but also on the unstructured meshes. Our interest in such implementations stems from the necessity to solve Eikonal equation not only in the plane, but also on manifolds. As shown in [46], if the manifold is a graph of some function $z = g(x, y)$, then the isotropic front propagation on this manifold corresponds to a more general (anisotropic) front propagation problem in the $x-y$ plane. The corresponding (more general) Hamilton-Jacobi PDE in the plane does not posses the *causality* property and that makes it much more difficult to solve using a single-pass method (see section 4.1). Hence, the following approach, introduced in [27], is preferable: the manifold is approximated by some triangulated mesh X and the Fast Marching Method (or Tsitsiklis' Algorithm) is applied to solve the Eikonal equation on that mesh.

Moreover, if we start with an anisotropic front propagation problem in the plane, it is sometimes possible to reformulate it as an isotropic problem on a certain manifold. The following "lifting to manifold" procedure was introduced in [48].

[6] Additionally, we note that we have extended Tsitsiklis' original Algorithm to unstructured meshes in R^2 and on manifolds. The splitting section techniques developed for the Fast Marching Method can also be used to implement Tsitsiklis' Algorithm on meshes with non-acute simplexes.

Suppose we are given a graph of a function $z = g(x, y)$, and attempt to solve the Eikonal equation $\|\nabla u\| = 1/F(x, y)$ on that manifold. To be clear, $F(x, y)$ gives the speed in the direction normal to the level line $u = constant$ on the manifold $z = g(x, y)$. Projecting down onto the $x - y$ plane, this translates into a particular static Hamilton-Jacobi equation on the plane. We now use this argument in reverse as follows. Consider any static Hamilton-Jacobi equation in the $x - y$ plane of the form

$$a(x, y)u_x^2 + b(x, y)u_x u_y + c(x, y)u_y^2 = g(x, y) \tag{3.17}$$

together with boundary conditions for u. Now suppose we can find functions $p(x, y)$, $q(x, y)$ and $F(x, y)$ such that $p_y = q_x$, $F(x, y) > 0$, and

$$\begin{aligned} a(x, y) &= (1 + q^2); \qquad b(x, y) = -2pq; \\ c(x, y) &= (1 + p^2); \qquad g(x, y) = (1 + p^2 + q^2)F^{-2}(x, y). \end{aligned}$$

It can be then shown (see [46]) that the solution of Eqn 3.17 can be obtained by solving the Eikonal equation $\|\nabla u\| = 1/F(x, y)$ on the manifold $z = g(x, y)$ where $f_x = p$ and $f_y = q$. Thus, for any static Hamilton-Jacobi equation of the form given by Eqn 3.17, if we can find functions p and q satisfying the above, then we can *construct* the surface $z = g(x, y)$, approximate it with a triangulated mesh, and then solve the straightforward Eikonal problem on the manifold.

As an example, we consider the following equation

$$\begin{aligned} (1 + \gamma \sin^2(2\pi x)\cos^2(2\pi y))u_x^2 - \tfrac{\gamma}{2}\sin(4\pi x)\sin(4\pi y)u_x u_y + (1 + \gamma\cos^2(2\pi x)\sin^2(2\pi y))u_y^2 = \\ = 1 + \gamma\cos^2(2\pi x)\sin^2(2\pi y) + \gamma\sin^2(2\pi x)\cos^2(2\pi y), \end{aligned}$$

where $\gamma = (.9\pi)^2$, $x, y \in [0, 1]$, and the boundary condition is $u(.5, .5) = 0$. We can find functions $p = .9\pi\cos(2\pi x)\sin(2\pi y)$, $q = .9\pi\sin(2\pi x)\cos(2\pi y)$ and $F = 1$ which satisfy our compatibility requirements, and then solve the Eikonal equation $\|\nabla u\| = 1$ on the surface $g(x, y) = .45\sin(2\pi x)\sin(2\pi y)$. Figure 3.4 shows the evolving front on the surface, and the solution to the original problem on the plane.

Whenever such a transformation is possible, the problem can be addressed using the single-pass methods for the Eikonal equation applied to a mesh approximating the manifold $z = g(x, y)$. Such examples can be very useful as the test cases for the general method for anisotropic problems. We can use the general method to solve the Hamilton-Jacobi PDE 3.17 in the plane and then compare them to the numerical solution produced

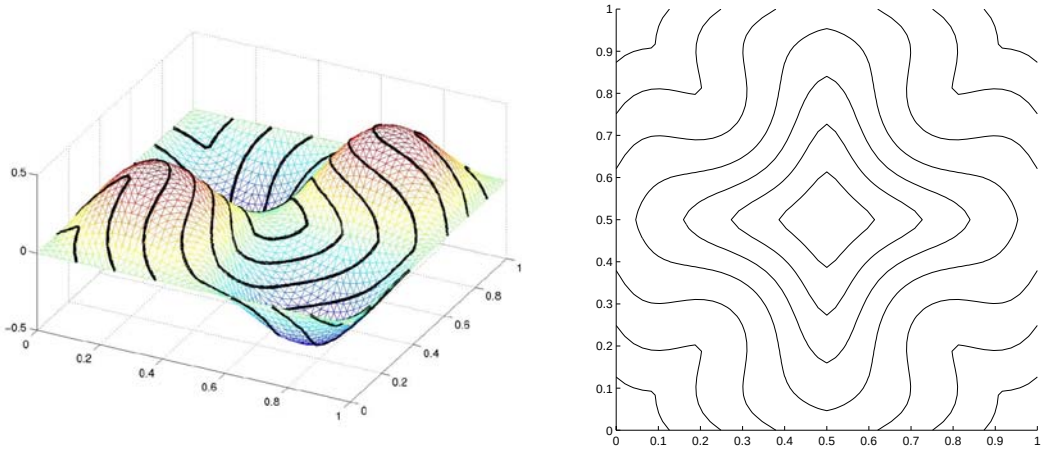


Figure 3.4: Using Fast Marching Method and "lifting-to-manifold" to solve an anisotropic static Hamilton-Jacobi equation.

by an Eikonal solver on the corresponding manifold. We will also see in Chapter 7 that the "lifting-to-manifold" technique can be quite useful in the context of our general single-pass methods in order to reduce the anisotropy coefficient Υ.

Chapter 4

Fast Method for the Anisotropic Problems: the Algorithm.

In this chapter we will first attempt to use Tsitsiklis' Algorithm to solve some more general (non-Eikonal) problems, we will then discuss the inherent reasons for the failure of such attempts, and, finally, we will produce a fast method for the general (anisotropic) Hamilton-Jacobi-Bellman PDE introduced in section 2.2.3. Our method will use Gonzales-Rofman control-theoretic discretization [24] as a building block and we will provide an optimality-type argument to motivate the use of the method. In Chapter 5 we will revisit this method to discuss the properties of the numerical solution $U(\boldsymbol{x})$ and to prove its convergence to the viscosity solution.

A sample anisotropic problem considered here corresponds to solving a "distance from the boundary" equation on some manifold $z = g(x, y)$. As we have seen in Chapter 2, the viscosity solution of a "distance from the boundary" PDE is also the value function of the corresponding min-time unit-speed optimal trajectory problem. In this formulation, a vehicle is moving with unit speed (and unit cost) on the manifold $z = g(x, y)$. To formulate an equivalent problem, we now consider a second vehicle moving in the $x - y$ plane, which is just a shadow[1] of the first one. Thus, its trajectories are the orthogonal projections onto the $x - y$ plane of the first vehicle's trajectories on the manifold. Since the first vehicle's velocity is a unit vector in the manifold's tangent plane, the length of its projection onto

[1] We assume that the Sun is at its zenith.

the $x-y$ plane yields the speed function for the shadow:

$$f(\boldsymbol{a}, x, y) = \left(1 + (\nabla g(x,y) \cdot \boldsymbol{a})^2\right)^{-\frac{1}{2}}, \tag{4.1}$$

where $\boldsymbol{a}$ is a vector of unit length in the $x-y$ plane.

Consider a plane $z = c_1 x + c_2 y$ for some vector $\boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$. If we solve on that plane the "distance from the origin" Eikonal equation

$$\|\nabla d\| = 1,$$
$$d(0,0) = 0,$$

then the level sets of d will be just the circles around the origin in that plane. If we project those circles orthogonally onto the $x-y$ plane, the obtained ellipses will be the level sets of the viscosity solution $u(\boldsymbol{x})$ of the equation

$$\min_{\boldsymbol{a} \in S_1} \{(\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}) f(\boldsymbol{x}, \boldsymbol{a})\} + 1 = 0, \qquad \boldsymbol{x} \in \Omega, \tag{4.2}$$

and the vehicle's speed function in the direction $\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ will be given by

$$f(\boldsymbol{a}, x, y) = \left(1 + (c_1 a_1 + c_2 a_2)^2\right)^{-\frac{1}{2}}.$$

This constitutes our simplest test problem for this chapter.

Remark 4.0.1. As shown in [46], this same problem can be viewed in the front propagation framework using the speed function

$$F(\boldsymbol{x}, \boldsymbol{n}) = \sqrt{\frac{(1+c_2^2)n_1^2 + (1+c_1^2)n_2^2 - 2c_2 c_1 n_1 n_2}{1 + c_1^2 + c_2^2}}.$$

The corresponding PDE for this speed function F is

$$\frac{(1+c_2^2)u_x^2(\boldsymbol{x}) + (1+c_1^2)u_y^2(\boldsymbol{x}) - 2c_2 c_1 u_x(\boldsymbol{x}) u_y(\boldsymbol{x})}{1 + c_1^2 + c_2^2} = 1, \tag{4.3}$$

which can be shown to be equivalent to the Hamilton-Jacobi-Bellman PDE 4.2. We will discuss the general correspondence between these two classes of problems in Chapter 6. Here we simply note that this sample problem can be used to test the direct anisotropic modifications of both Tsitsiklis' Algorithm and Sethian's Fast Marching Method (using equation 4.3).

4.1 Characteristics vs. gradients.

First of all, as demonstrated in section 2.2.5, we can switch from the min-time to the min-cost optimal trajectory problem by choosing $K(\boldsymbol{x}, \boldsymbol{a}) = \frac{1}{f(\boldsymbol{x},\boldsymbol{a})}$. Recall that the Gonzales-Rofman scheme is valid in the general non-Eikonal (i.e., anisotropic) problems. Thus, it seems natural to use Tsitsiklis' single-pass algorithm (derived for the Eikonal equation) to de-couple the system arising in this more general case.

We begin by noting that the algorithm defined in section 2.2.10 can be applied without any changes at all, except that the dependence of K (or f) upon the direction $\boldsymbol{a}$ will now be present in the update-from-a-single-simplex formula :

$$V_s(\boldsymbol{x}) = \min_{\zeta \in [0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \boldsymbol{a}_\zeta)} + \zeta U(\boldsymbol{x}_{\boldsymbol{s},\mathbf{1}}) + (1-\zeta) U(\boldsymbol{x}_{\boldsymbol{s},\mathbf{2}}) \right\}. \tag{4.4}$$

What happens when this algorithm is used to compute the expansion of ellipse (equation 4.2)? In Figure 4.1 we show the level sets of the numerical solution U obtained by this method for two different expanding ellipses. The first contour plot corresponds to the vector $\boldsymbol{c} = \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$. The numerical solution converges to the value function $u(\boldsymbol{x})$ and is first order accurate as the grid size tends to 0. (We will return to this example later when we discuss fast methods relying on a particular grid orientation in section 7.3.1.) The second contour plot corresponds to the vector $\boldsymbol{c} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. In this case, it is obvious that $U(\boldsymbol{x})$ does not approximate the viscosity solution very well. Nor does it improve under a grid refinement[2].

In order to understand what is different in the second example, we have to recall that both Tsitsiklis' Algorithm and Sethian's Fast Marching Method are fundamentally dependent on the *causality property* (2.2.19) of the Eikonal equation. Each of these single-pass methods is based on the observation that a certain discretization also possesses a similar *causality property*[3]. The *causality* results from the fact that the characteristics of the Eikonal equation coincide with the gradient lines of its viscosity solution u. However, for the anisotropic problems this property does not hold. When the characteristic and gradient

[2]The Fast Marching Method, used on a Cartesian grid with a suitable upwind finite difference discretization of the equation 4.3, produces numerical solutions qualitatively similar to those displayed in Figure 4.1.

[3]The Fast Marching Method uses this property to efficiently de-couple the upwind finite difference discretization presented in [38]; Tsitsiklis' algorithm does the same with the control-theoretic discretization presented in [24].

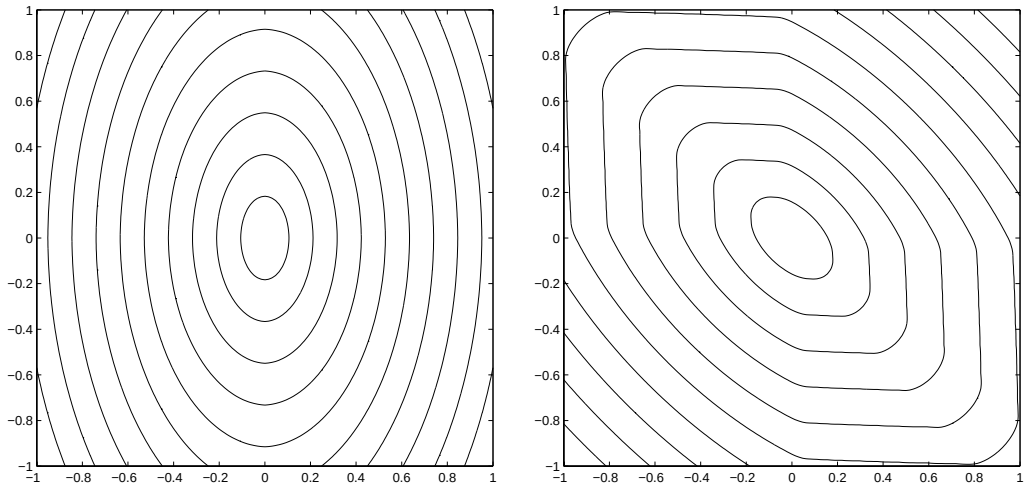


Fig.4.1A Fig.4.1B

Figure 4.1: Ellipse expansion computed by Tsitsiklis' Algorithm. Both computations performed on a 129×129 uniform Cartesian grid.

directions are different, the simplex $\boldsymbol{x x_j x_k}$ may contain the characteristic for the point $\boldsymbol{x}$, even if the gradient $\nabla u(\boldsymbol{x})$ is not pointing from that simplex (see Figure 4.2). Thus, no matter how small that simplex is, it is still possible that $u(\boldsymbol{x}) < u(\boldsymbol{x_j})$. This is precisely why both Sethian's Fast Marching Method and Tsitsiklis' Algorithm cannot be directly applied in the general anisotropic (non-Eikonal) case: it is no longer possible to de-couple the system by computing/accepting the mesh points in the ascending order.

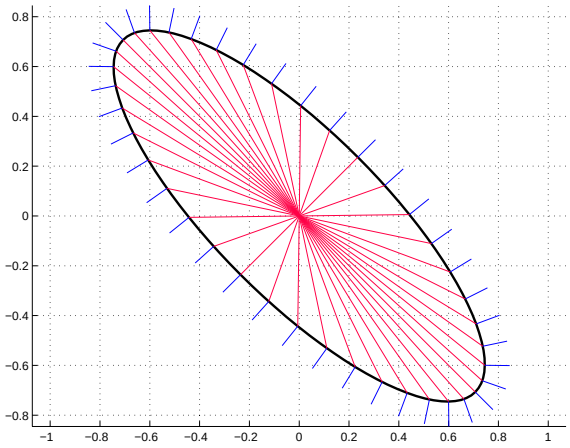


Figure 4.2: The characteristics and the gradient directions for an expanding ellipse.

Remark 4.1.1. By comparing Figures 4.1B and 4.2, we observe that the isotropic single-pass method fails exactly at those points where the gradient line and the characteristic do not lie in the same coordinate quadrant (or, more generally, in the same simplex - the quadrants are used because the numerical solution in Figure 4.1 is computed on a Cartesian grid). This is an intrinsic problem: to produce the numerical solution efficiently, both of the above single-pass methods attempt to compute $U(\boldsymbol{x})$ in the ascending order (i.e., from the simplex containing $(-\nabla u)$), whereas, in order to maintain the upwinding, $U(\boldsymbol{x})$ has to be computed from the simplex containing the characteristic.

4.2 Causality in the Hamilton-Jacobi-Bellman PDE

A different (weaker) *causality property* for the more general Hamilton-Jacobi-Bellman results from Bellman's optimality principle (section 2.2). Since the characteristics of that PDE are, in fact, the optimal trajectories of the corresponding control problem, we know that the value function u is strictly increasing along the characteristics.
An alternative formulation of Bellman's principle will be even more useful for building the single-pass method:

Let $u(\boldsymbol{x})$ be the value function for the anisotropic min-cost optimal trajectory problem defined on Ω in section 2.2. We will use the notation $K_{\hat{\boldsymbol{x}}}(\boldsymbol{x})$ for the minimum cost required to reach the point $\hat{\boldsymbol{x}}$ starting from the point $\boldsymbol{x}$. More rigorously, let $\mathcal{A}(\hat{\boldsymbol{x}})$ be the set of all controls $\boldsymbol{a}(\cdot)$ such that the corresponding trajectory

$$\begin{aligned} \boldsymbol{y}'(t) &= \boldsymbol{a}(t), \\ \boldsymbol{y}(0) &= \boldsymbol{x} \end{aligned}$$

passes through the point $\hat{\boldsymbol{x}}$ at some time $\tau(\boldsymbol{a}(\cdot))$ (i.e., $\boldsymbol{y}(\tau(\boldsymbol{a}(\cdot)) = \hat{\boldsymbol{x}}$). We can now formally define

$$K_{\hat{\boldsymbol{x}}}(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)\in\mathcal{A}(\hat{\boldsymbol{x}})} \left\{ \int_0^{\tau(\boldsymbol{a}(\cdot))} K(\boldsymbol{y}(s), \boldsymbol{a}(s)) ds \right\}.$$

Optimality Principle: Consider a simple closed curve $\Gamma \subset \Omega \backslash \partial\Omega$. It is easy to show for every $\boldsymbol{x}$ inside Γ that

$$u(\boldsymbol{x}) = \inf_{\hat{\boldsymbol{x}} \in \Gamma} \left\{ K_{\hat{\boldsymbol{x}}}(\boldsymbol{x}) + u(\hat{\boldsymbol{x}}) \right\}. \tag{4.5}$$

Remark 4.2.1. The proof of this statement is very similar to the proof of Lemma 2.2.4. This variant of the optimality principle is different in that the infimum is taken not after

some fixed time τ, but whenever the trajectory intersects the curve Γ. Because of the properties of the running cost K and by the continuity of Γ, that infimum is actually a minimum achieved at some point $\tilde{\boldsymbol{x}}$:

$$u(\boldsymbol{x}) = K_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}) + u(\tilde{\boldsymbol{x}}).$$

Thus, knowing u on Γ is sufficient to evaluate u at any point inside Γ. Moreover, if Γ is a level set of $u(\boldsymbol{x})$, then, by the Lemma 2.2.11, we know that

$$\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq d_1 \Upsilon, \tag{4.6}$$

where $\Upsilon = \frac{K_2}{K_1}$ and d_1 is the distance from $\boldsymbol{x}$ to Γ. The last observation necessary to construct a computational algorithm is that, if d_1 is small relative to the size of Γ, then the optimal cost $K_{\tilde{\boldsymbol{x}}}(\boldsymbol{x})$ cannot be much smaller than the cost of the trajectory running directly from $\boldsymbol{x}$ to $\tilde{\boldsymbol{x}}$. We are now ready to formally state our control-theoretic Ordered Upwinding Method[4].

4.3 Control-theoretic fast method.

Ab exterioribus ad interiora... *"From the outside - inwards..."*

ST. AUGUSTINE (354-430 AD) writer, philosopher
An excerpt from his *Numerical Methods for Boundary Value Problems*

Consider an unstructured triangulated mesh X of diameter h (i.e., if the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent then $\|\boldsymbol{x_j} - \boldsymbol{x_k}\| \leq h$).

Let $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ be two adjacent mesh points. Define the upwinding approximation for $U(\boldsymbol{x})$ from the simplex $\boldsymbol{x_j x x_k}$:

$$V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) = \min_{\zeta \in [0,1]} \left\{ \tau(\zeta) K(\boldsymbol{x}, \boldsymbol{a}_\zeta) + \zeta U(\boldsymbol{x_j}) + (1-\zeta) U(\boldsymbol{x_k}) \right\}, \tag{4.7}$$

where $\tau(\zeta) = \|(\zeta \boldsymbol{x_j} + (1-\zeta)\boldsymbol{x_k}) - \boldsymbol{x}\|$, and $\boldsymbol{a}_\zeta = \frac{(\zeta \boldsymbol{x_j} + (1-\zeta)\boldsymbol{x_k}) - \boldsymbol{x}}{\tau(\zeta)}$.

Remark 4.3.1. The above update formula is basically the same as the upwind formula for simplex s in (2.27). The difference is that $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is defined even when $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are not adjacent to $\boldsymbol{x}$.

[4]Since Γ generally is not a level set of u, the logic of the method is more subtle and cannot really be based on Lemma 2.2.11. Instead, it relies on Lemma 2.2.12, which provides a weaker version of (4.6), but for any Γ "well-resolved" by an underlying mesh X.

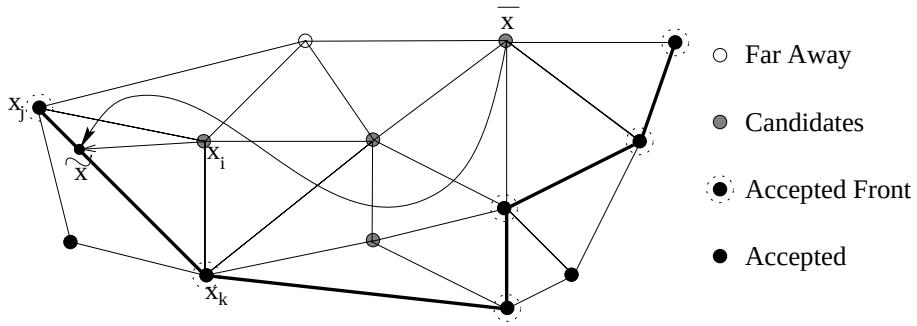


Figure 4.3: The *AcceptedFront* and the *Considered* mesh points. The optimal trajectory for $\bar{\boldsymbol{x}}$ cannot intersect AF too far away from $\bar{\boldsymbol{x}}$, for if $\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| > h\frac{F_2}{F_1}$, then $u(\boldsymbol{x_i}) < u(\bar{\boldsymbol{x}})$.

As before, mesh points are divided into three classes (*Far*, *Considered*, *Accepted*). The *AcceptedFront* is defined as a set of *Accepted* mesh points, which are adjacent to some not-yet-accepted (i.e., *Considered*) mesh points. Define the set AF of the line segments $\boldsymbol{x_j x_k}$, where $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent mesh points on the *AcceptedFront*, such that there exists a *Considered* mesh point $\boldsymbol{x_i}$ adjacent to both $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$. For each *Considered* mesh point $\boldsymbol{x}$ we define the "near front" as the part of AF "relevant to $\boldsymbol{x}$":

$$\mathrm{NF}(\boldsymbol{x}) = \left\{ \boldsymbol{x_j x_k} \in AF \;\middle|\; \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_j x_k} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \le h\frac{F_2}{F_1} \right\}.$$

1. Start with all the mesh points in Far.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{\boldsymbol{x_j x_k} \in \mathrm{NF}(\boldsymbol{x})} V_{\boldsymbol{x_j}, \boldsymbol{x_k}}(\boldsymbol{x}). \tag{4.8}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$) and update the *AcceptedFront*.
6. Move the Far mesh points adjacent to $\bar{\boldsymbol{x}}$ into *Considered*.
7. Recompute the value for all the *Considered* $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$

$$V(\boldsymbol{x}) := \min\left\{ V(\boldsymbol{x}), \min_{\bar{\boldsymbol{x}} \boldsymbol{x_i} \in \mathrm{NF}(\boldsymbol{x})} V_{\bar{\boldsymbol{x}}, \boldsymbol{x_i}}(\boldsymbol{x}) \right\}. \tag{4.9}$$

8. If *Considered* is not empty then go to 4.

What I tell you three times is true.

LEWIS CARROLL (1832-98)

Motivation: Assume that $U(\boldsymbol{x}) = u(\boldsymbol{x})$ for any $\boldsymbol{x} \in AF$. Suppose that a *Considered* $\bar{\boldsymbol{x}}$ is such that[5] $u(\bar{\boldsymbol{x}}) = \min_{\boldsymbol{x} \notin Accepted} u(\boldsymbol{x})$. Let $\boldsymbol{a_1}(\cdot)$ be the optimal control for $\bar{\boldsymbol{x}}$ and let $\boldsymbol{y}(t)$ be the optimal trajectory corresponding to it. By the Lemma 2.2.12, the optimal trajectory $\boldsymbol{y}(t)$ will intersect AF at some point $\tilde{\boldsymbol{x}}$ such that $\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq h\frac{K_2}{K_1}$. Therefore, $\tilde{\boldsymbol{x}} \in \mathrm{NF}(\bar{\boldsymbol{x}})$; see Figure 4.3. We note that $\tilde{\boldsymbol{x}} = \zeta \boldsymbol{x_j} + (1-\zeta)\boldsymbol{x_k}$ for some $\zeta \in [0,1]$ and for some line segment $\boldsymbol{x_j x_k} \in \mathrm{NF}(\bar{\boldsymbol{x}})$. Using some additional assumptions about the regularity of (sub)optimal trajectories, we could also show that $u(\bar{\boldsymbol{x}}) = u(\tilde{\boldsymbol{x}}) + K(\bar{\boldsymbol{x}}, \frac{\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\tau(\zeta)})\tau(\zeta) + O(h^2)$. If ∇u existed everywhere on the segment $\boldsymbol{x_j x_k}$, we could also show that $u(\bar{\boldsymbol{x}}) = \zeta u(\boldsymbol{x_j}) + (1-\zeta)u(\boldsymbol{x_k}) + K(\bar{\boldsymbol{x}}, \frac{\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\tau(\zeta)})\tau(\zeta) + O(h^2)$. Finally, recalling that we assumed $U(\boldsymbol{x}) = u(\boldsymbol{x})$ for every mesh point $\boldsymbol{x} \in AF$, we see that

$$U(\bar{\boldsymbol{x}}) \approx u(\bar{\boldsymbol{x}}) \approx \zeta U(\boldsymbol{x_j}) + (1-\zeta)U(\boldsymbol{x_k}) + K\left(\bar{\boldsymbol{x}}, \frac{\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}}{\tau(\zeta)}\right)\tau(\zeta) + O(h^2).$$

Remark 4.3.2. A careful construction modeled after the above argument could, theoretically, serve as a "step of induction" in proving that $|U(\boldsymbol{x}) - u(\boldsymbol{x})| = O(h)$. This is a reasonable assumption since the optimal trajectory probably passes through $O(1/h)$ simplexes - if it wanders too much it cannot be optimal. Unfortunately, the additional difficulty with this argument is that $u(\boldsymbol{x_j}) + (1-\zeta)u(\boldsymbol{x_k}) = u(\tilde{\boldsymbol{x}}) + O(h^2)$ only if ∇u is everywhere defined on $\boldsymbol{x_j x_k}$. In the presence of a shock, the error in that approximation could, in fact, be $O(h)$. It is possible to show that, for the problem formulated in section 2.2, an optimal trajectory will not intersect the shock[6], but the "approximate optimal trajectory" might. Showing that we do not have to interpolate across the shock "too often" would be even a bigger challenge. This is the main reason for our choosing a different (function-analytic) approach to prove the convergence in Chapter 5.

[5] As prescribed by the algorithm, $\bar{\boldsymbol{x}}$ is selected for having the smallest V and not the smallest u (we would not be computing the numerical solution if we knew $u(\boldsymbol{x})$ already!). This argument could be made rigorous by employing $v(\boldsymbol{x})$, a continuous analogue of $V(\boldsymbol{x})$. We choose to ignore this issue since the above reasoning is only motivational. The rigorous proof of convergence to the viscosity solution will be provided in Chapter 5.

[6] This follows from the entropy requirement that the characteristics can run into shocks, but cannot emerge from shocks, i.e., no new information is created at the shocks.

We note that the resulting algorithm

- is "single-pass", since it produces the numerical solution U in $O\left(\left(\frac{K_2}{K_1}\right)^2 M \log(M)\right)$ steps:
  - total of M points to *Accept*;
  - every time a mesh point is accepted there are at most $\left(\frac{K_2}{K_1}\right)^2$ *Considered* points to re-evaluate;
  - a necessity to maintain an ordering of *Considered* based on V accounts for the factor of $\log(M)$.
- produces the numerical solution U which converges to u as the diameter of the mesh tends to zero (see the proof in Chapter 5);
- is at most first order accurate;
- works equally well on acute and non-acute triangulated meshes (no additional splitting section construction is required for the non-acute meshes);
- is applicable for a general anisotropic optimal trajectory problem described in section 2.2.

An extension of this method to R^n and manifolds is straightforward, since the update formula 4.7 can be easily generalized for these cases.

Remark 4.3.3 (Three comments on the computational complexity).

1. In the above complexity analysis the calculation of an upwind-update-from-a-single-simplex value $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ was counted as a single operation. Of course, in practice, for a general anisotropic optimal trajectory problem, this value can be computed only approximately and will require a number of iterations depending on the required precision. This, in general, is unavoidable, and such a calculation is normally considered to be a single operation in the analysis of computational complexity for the control-theoretic algorithms. We note that the optimization problem solved to compute $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is **local** (i.e., $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ can be computed independent from any other $V_{\boldsymbol{x_i},\boldsymbol{x_m}}(\boldsymbol{x_l})$) and, thus, should not be confused with the iterations necessary to solve

the coupled system of non-linear equations (2.27) simultaneously.

More details on algorithmic efficiency analysis can be found in Chapter 7.

2. As we will show in Chapter 5, AF can be considered as an approximation for a level set of U. Thus, if the mesh diameter h is sufficiently small, then the number of *Considered* points which have to be updated after each acceptance becomes closer to $\left(\frac{K_2}{K_1}\right)$, since the *Considered* points are immediately adjacent to AF. Thus, as h decreases, the computational complexity of the method tends to $O\left(\left(\frac{K_2}{K_1}\right) M \log(M)\right)$.

3. If the problem were formulated in R^n, the complexity of the corresponding algorithm would be $O\left(\left(\frac{K_2}{K_1}\right)^{n-1} M \log(M)\right)$, where M still is the total number of mesh points.

Remark 4.3.4 (A comment on the rate of convergence).
Our proof of convergence in Chapter 5 does not provide an estimate for the rate of convergence. We believe that this method is first order; this belief is based on the first order accuracy of the approximations behind Gonzales-Rofman discretization (used to calculate $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$) and is confirmed by the numerical evidence.
Based on our numerical experiments, we note that for sufficiently small h, $\|U-u\|_\infty = O(\frac{K_2}{K_1}h)$. This is not surprising, since $\frac{K_2}{K_1}h$ is the largest distance, over which the first order accurate approximation might be performed, when $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is computed.

Remark 4.3.5 (A comment on mesh degeneracy).
It is not surprising that the acuteness of simplexes in X is not required. After all, the algorithm uses the mesh connectivity only to determine what becomes *Considered*, when a new mesh point is *Accepted*. All of the upwind-update-from-a-single-simplex values $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ are computed from the simplexes defined by the position of $AcceptedFront$ rather than from the simplexes present in X.

Nevertheless, in order to prove the convergence of $U(\boldsymbol{x})$ to the viscosity solution, we will have to assume that the mesh X cannot be arbitrarily degenerate. Namely, we will assume that, if h is the diameter of X and h_{min} is the smallest triangle height in X, then the ratio $\eta = h/h_{min}$ is bounded for all sufficiently small h. See the proof of Lemma 5.1.6 for details.

Remark 4.3.6 (A comment on the order of *Acceptance*).
Unlike in Sethian's Fast Marching Method or in Tsitsiklis' Algorithm, in the above method

the mesh points **are not** *Accepted* in the order of increasing U. As it was pointed out in section 4.1, for the anisotropic optimal trajectory problems the fact that the characteristic for $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x x_j x_k}$ does not mean that the gradient is pointing from that simplex. Thus, it is entirely possible that $U(\boldsymbol{x}) \leq V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) < U(\boldsymbol{x_j})$. Nevertheless, we will show in Lemma 5.1.4 that the order of *Acceptance* is monotone, albeit in a much weaker sense than for the single-pass Eikonal solvers.

4.3.1 What are we de-coupling here?

Define the extended set of neighbors

$$N_K(\boldsymbol{x}) = \left\{ \boldsymbol{x_1 x_2} \in X \mid \boldsymbol{x_1} \text{ and } \boldsymbol{x_2} \text{ are adjacent and } \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_1 x_2} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq h\frac{K_2}{K_1} \right\}$$

Note that if we substitute $\mathrm{NF}(\boldsymbol{x})$ by $N_K(\boldsymbol{x})$, the formula 4.8 becomes

$$U(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in N_K(\boldsymbol{x})} V_{\boldsymbol{x_1 x_2}}(\boldsymbol{x}). \tag{4.10}$$

This is a discretized version of Hamilton–Jacobi–Bellman Eqn (2.15); more precisely, it is an "extended" version of Gonzales-Rofman scheme 2.27 and it is easy to show that its solution U converges to the viscosity solution u. The Eqn 4.10 can be solved by successive approximation techniques described in [24], for example. However, a Dijkstra-like single-pass algorithm cannot be used to find U since we need to consider all possible directions of motion for the vehicle starting at the point $\boldsymbol{x}$ (i.e., $U(\boldsymbol{x})$ might potentially depend upon $U(\boldsymbol{y})$ for all $\boldsymbol{y} \in N_K(\boldsymbol{x})$, including the values $U(\boldsymbol{y}) > U(\boldsymbol{x})$). Therefore, the formula 4.8 can be interpreted as an upwinding analogue of Eqn 4.10.

Remark 4.3.7 (A comment on a peculiarity of the method:).
The above comparison with an extended Gonzales-Rofman scheme is just an analogue - not an equivalence. The numerical values produced by executing the above algorithm will be different from those obtained by solving the coupled system 4.10.
In fact, this is a somewhat peculiar characteristic of our method. The majority of numerical methods for static PDEs are based on

1. deriving some consistent discretized version of the equation,
2. proving that there exists a numerical solution for that discretized equation,
3. deriving an algorithm to find that numerical solution,

4. and proving that this numerical solution converges to the desired solution of the original PDE as the mesh (or grid) is refined.

As of right now, we do not know of any natural discretized version of the Hamilton-Jacobi-Bellman PDE, which would be exactly satisfied by the numerical solution $U(\boldsymbol{x})$ produced by the Ordered Upwinding Method described in this section. Therefore, the second and the third stages outlined above are irrelevant, since $U(\boldsymbol{x})$ is defined constructively (i.e., by an algorithm to compute it). Unfortunately, since we cannot rely on properties of a discretized equation, the proof of convergence (Chapter 5) has to rely on the properties of the algorithm itself and, thus, becomes a real exercise in hand-to-hand calculus combat.

Chapter 5

Fast Method for the Anisotropic Problems: the Proof of Convergence.

> The pursuit of pretty formulas and neat theorems can no doubt quickly degenerate into a silly vice, but so can the quest for austere generalities which are so very general indeed that they are incapable of application to any particular.
>
> ERIC TEMPLE BELL(1883 - 1960)

For the duration of this chapter we will assume that the numerical solution $U(\boldsymbol{x})$ is computed for each $x \in X$ using the Ordered Upwinding Method described in section 4.3. For the points $\boldsymbol{x} \in \Omega \backslash X$, $U(\boldsymbol{x})$ is defined by linear interpolation:

If $\boldsymbol{x}$ is inside Ω, but is not a mesh point, then it lies in some simplex $\boldsymbol{x_1 x_2 x_3}$. In that case, $\exists \zeta_1, \zeta_2, \zeta_3 \geq 0$ such that

$$\begin{aligned} \zeta_1 + \zeta_2 + \zeta_3 &= 1, \\ \zeta_1 \boldsymbol{x_1} + \zeta_2 \boldsymbol{x_2} + \zeta_3 \boldsymbol{x_3} &= \boldsymbol{x}. \end{aligned}$$

The value at $\boldsymbol{x}$ is defined to be $U(\boldsymbol{x}) = \zeta_1 U(\boldsymbol{x_1}) + \zeta_2 U(\boldsymbol{x_2}) + \zeta_3 U(\boldsymbol{x_3})$.

Suppose h_{min} is the smallest triangle height in the mesh X. We will use the constant $\eta = \frac{h}{h_{min}}$ to characterize the degree of "degeneracy" of the mesh X.

5.1 Properties of the numerical solution

> I have had my results for a long time: but I do not yet know how I am to arrive at them.
>
> KARL FRIEDRICH GAUSS (1777-1855)

In this section we prove several Lemmas, which are later used in our proof of convergence. The properties of the numerical solution U demonstrated in this section are similar to the properties of the value function for the optimal trajectory problem (section 2.2.2).

5.1.1 Is NF($\boldsymbol{x}$) big enough?

Suppose the mesh point $\bar{\boldsymbol{x}}$ is about to be *Accepted*. This means that $V(\bar{\boldsymbol{x}}) = \min_{\boldsymbol{x} \in Considered} V(\boldsymbol{x})$ and therefore $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$.

Lemma 5.1.1. *For every Considered mesh point $\boldsymbol{x}$ define*

$$W(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in AF} \min_{\zeta \in [0,1]} \left\{ K(\boldsymbol{x}, \boldsymbol{a}) \tau(\zeta) + (\zeta U(\boldsymbol{x_1}) + (1-\zeta) U(\boldsymbol{x_2})) \right\}, \tag{5.1}$$

where $\tau(\zeta) = \|(\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2}) - \boldsymbol{x}\|$, *and* $\boldsymbol{a} = \frac{\zeta(\boldsymbol{x_1} - \boldsymbol{x}) + (1-\zeta)(\boldsymbol{x_2} - \boldsymbol{x})}{\tau(\zeta)}$.
Claim: $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) = W(\bar{\boldsymbol{x}})$.

Proof. First of all, $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$ simply because $\bar{\boldsymbol{x}}$ is about to be *Accepted*.
Recall that $V(\boldsymbol{x})$ for every *Considered* mesh point $\boldsymbol{x}$ is computed by the formula 4.8 :

$$V(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in \mathrm{NF}(\boldsymbol{x})} \min_{\zeta \in [0,1]} \left\{ K(\boldsymbol{x}, \boldsymbol{a}) \tau(\zeta) + (\zeta U(\boldsymbol{x_1}) + (1-\zeta) U(\boldsymbol{x_2})) \right\},$$

where $\tau(\zeta) = \|(\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2}) - \boldsymbol{x}\|$, $\boldsymbol{a} = \frac{\zeta(\boldsymbol{x_1} - \boldsymbol{x}) + (1-\zeta)(\boldsymbol{x_2} - \boldsymbol{x})}{\tau(\zeta)}$, and NF($\boldsymbol{x}$) is the part of the *AcceptedFront* "relevant to $\boldsymbol{x}$":

$$\mathrm{NF}(\boldsymbol{x}) = \left\{ \boldsymbol{x_1 x_2} \in AF \mid \exists \tilde{\boldsymbol{x}} \text{ on the line segment } \boldsymbol{x_1 x_2} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq h \frac{K_2}{K_1} \right\}.$$

Since NF($\boldsymbol{x}$) $\subset AF$ we immediately see that for any *Considered* mesh point $\boldsymbol{x}$

$$V(\boldsymbol{x}) \geq W(\boldsymbol{x}). \tag{5.2}$$

Let $\boldsymbol{x_1 x_2} \in AF$ and $\zeta \in [0,1]$ be such that the minimum in the formula 5.1 is attained, i.e., if $\hat{\boldsymbol{x}} = (\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2})$ then

$$W(\bar{\boldsymbol{x}}) = K(\bar{\boldsymbol{x}}, \frac{\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}) \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + (\zeta U(\boldsymbol{x_1}) + (1-\zeta) U(\boldsymbol{x_2})). \tag{5.3}$$

Let $\boldsymbol{x_3}$ be the *Considered* mesh point adjacent to both $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. Then $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) \leq V(\boldsymbol{x_3})$ since $\bar{\boldsymbol{x}}$ is about to be accepted.
$V(\boldsymbol{x_3})$ is also computed by the formula 4.8; thus,

$$V(\boldsymbol{x_3}) \;\leq\; K(\boldsymbol{x_3}, \frac{\hat{\boldsymbol{x}} - \boldsymbol{x_3}}{\|\hat{\boldsymbol{x}} - \boldsymbol{x_3}\|})\|\hat{\boldsymbol{x}} - \boldsymbol{x_3}\| + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2})) \;\leq$$
$$\leq\; K_2 h + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,. \tag{5.4}$$

Inequality is the cause of all local movements.

Leonardo da Vinci (1452-1519)

Combining the inequalities 5.2, 5.3, and 5.4 we obtain

$$K_1\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2})) \;\leq\; W(\bar{\boldsymbol{x}}) \;\leq\; V(\bar{\boldsymbol{x}}) \;\leq$$
$$\leq\; V(\boldsymbol{x_3}) \;\leq\; K_2 h + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,,$$

which implies $\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq h\frac{K_2}{K_1}$. Therefore, $\boldsymbol{x_1 x_2} \in \mathrm{NF}(\bar{\boldsymbol{x}})$ and $W(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$. □

5.1.2 Uniform upper bound.

An expert is the one who predicts the job will take the longest and cost the most.

Arthur Bloch

Lemma 5.1.2 (Uniform upper bound). *If Ω is convex and $d(\boldsymbol{x})$ is the distance from $\boldsymbol{x} \in \Omega$ to the boundary $\partial\Omega$, then*

$$U(\boldsymbol{x}) \leq d(\boldsymbol{x})K_2 + q_2. \tag{5.5}$$

Proof. If $\boldsymbol{x} \in \partial\Omega$ then the inequality trivially holds, since $0 \leq q(\boldsymbol{x}) \leq q_2$.
If $\boldsymbol{x}$ is a mesh point inside Ω, we prove the lemma by induction: assume that the inequality 5.5 holds for all the mesh points, which are on the *AcceptedFront* just before $\boldsymbol{x} = \bar{\boldsymbol{x}}$ is *Accepted*. Consider a (possibly non-unique) shortest path from $\bar{\boldsymbol{x}}$ to the boundary. By the properties of the distance function $d(\cdot)$, that shortest path is a straight line. Moreover,

suppose that line intersects the segment of *AcceptedFront* $\boldsymbol{x_1 x_2} \in AF$ at the point $\hat{\boldsymbol{x}} = (\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2})$. It is trivial to show that $d(\bar{\boldsymbol{x}}) = d(\hat{\boldsymbol{x}}) + \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|$ [1]. Using Lemma 5.1.1,

$$U(\bar{\boldsymbol{x}}) \;=\; W(\bar{\boldsymbol{x}}) \;\leq\; K(\bar{\boldsymbol{x}}, \frac{\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|})\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,.$$

Based on the assumption of induction,

$$U(\bar{\boldsymbol{x}}) \;\leq\; K_2\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + \zeta(K_2 d(\boldsymbol{x_1}) + q_2) + (1-\zeta)(K_2 d(\boldsymbol{x_2}) + q_2).$$

By the convexity of Ω, the distance-to-boundary function $d(\boldsymbol{x})$ is concave and $d(\hat{\boldsymbol{x}}) \geq \zeta d(\boldsymbol{x_1}) + (1-\zeta)d(\boldsymbol{x_2})$. Therefore,

$$U(\bar{\boldsymbol{x}}) \;\leq\; K_2(\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + d(\hat{\boldsymbol{x}})) + q_2 \;=\; K_2 d(\bar{\boldsymbol{x}}) + q_2,$$

which completes the proof by induction. (The base of induction is obvious, since only the mesh points on the boundary $\partial\Omega$ are already *Accepted* when the algorithm starts.)
If $\boldsymbol{x}$ is inside Ω, but is not a mesh point, then it lies in some simplex $\boldsymbol{x_1 x_2 x_3}$. In that case $\exists \zeta_1, \zeta_2, \zeta_3 \geq 0$ such that

$$\begin{aligned} \zeta_1 + \zeta_2 + \zeta_3 &= 1, \\ \zeta_1 \boldsymbol{x_1} + \zeta_2 \boldsymbol{x_2} + \zeta_3 \boldsymbol{x_3} &= \boldsymbol{x}. \end{aligned}$$

The value at $\boldsymbol{x}$ is defined to be $U(\boldsymbol{x}) = \zeta_1 U(\boldsymbol{x_1}) + \zeta_2 U(\boldsymbol{x_2}) + \zeta_3 U(\boldsymbol{x_3})$. Once again, using the concavity of the distance function,

$$\begin{aligned} U(\boldsymbol{x}) \;\leq\; & \zeta_1(K_2 d(\boldsymbol{x_1}) + q_2) + \zeta_2(K_2 d(\boldsymbol{x_2}) + q_2) + \zeta_3(K_2 d(\boldsymbol{x_3}) + q_2) \;\leq \\ & \leq\; q_2 + K_2\left(\zeta_1 d(\boldsymbol{x_1}) + \zeta_2 d(\boldsymbol{x_2}) + \zeta_3 d(\boldsymbol{x_3})\right) \;\leq\; K_2 d(\boldsymbol{x}) + q_2. \end{aligned}$$

□

Remark 5.1.3. The obtained bound is "uniform" since it is independent of the diameter h of the mesh X. We also note that a uniform upper bound on U can be derived even for a non-convex Ω assuming that η remains bounded and the boundary $\partial\Omega$ is adequately represented by the mesh as h tends to zero.

[1] Recall that the distance function is the viscosity solution of the Eikonal equation $\|d(\boldsymbol{x})\| = 1$ with the zero boundary conditions on $\partial\Omega$.

5.1.3 Relaxed monotonicity of the *Accepted*.

The single-pass methods for the Eikonal equation discussed in sections 2.2.10 and 3.4.2 were based on the *causality* property of the underlying discretizations. The ordering stemming from that causality was very simple: "$\boldsymbol{x_i}$ is *Accepted* after $\boldsymbol{x_j}$" implied "$U(\boldsymbol{x_i}) \geq U(\boldsymbol{x_j})$". Unfortunately, this implication is false for the method described in Chapter 4 for the more general Hamilton-Jacobi-Bellman PDE (2.15). As discussed in section 4.1, the gradient and the characteristic directions in that case are different. Thus, even if $U(\boldsymbol{x_i})$ was computed by an upwinding formula from the simplex $\boldsymbol{x_i x_j x_k}$ based on $U(\boldsymbol{x_j})$ and $U(\boldsymbol{x_k})$, it is still possible that $U(\boldsymbol{x_i}) < U(\boldsymbol{x_j})$.

However, a weaker monotonicity property can still be formulated, based on the evolution of AF during the computation. Recall that AF is defined as the set of the line segments $\boldsymbol{x_j x_k}$, where $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent mesh points on the $AcceptedFront$, s.t. there exists a *Considered* mesh point $\boldsymbol{x_i}$ adjacent to both $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$. Define U^{AF}_{min} (and U^{AF}_{max}) as the min (max) value of U on the set AF. Note that, since U is defined by the linear interpolation, both U^{AF}_{min} and U^{AF}_{max} are attained at the mesh points.
The following definitions are useful for discussing the evolution of $AcceptedFront$:

- $AF_{\bar{\boldsymbol{x}}}$ is the state of AF **immediately before** $\bar{\boldsymbol{x}}$ is *Accepted*.
- $U^{AF_{\bar{\boldsymbol{x}}}}_{min}$ and $U^{AF_{\bar{\boldsymbol{x}}}}_{max}$ are the minimum and maximum values of U on $AF_{\bar{\boldsymbol{x}}}$.
- $AF^{\bar{\boldsymbol{x}}}$ is the state of AF **immediately after** $\bar{\boldsymbol{x}}$ is *Accepted*.
- $U^{AF^{\bar{\boldsymbol{x}}}}_{min}$ and $U^{AF^{\bar{\boldsymbol{x}}}}_{max}$ are the minimum and maximum values of U on $AF^{\bar{\boldsymbol{x}}}$.

Lemma 5.1.4 (Monotonicity of AF's evolution.). *Suppose h_{min} is the smallest triangle height in the triangulated mesh X on Ω. Then the following weak monotonicity results hold for the numerical solution U:*

(i)

$$U^{AF_{\bar{\boldsymbol{x}}}}_{min} + h_{min}K_1 :\leq U(\bar{\boldsymbol{x}}) \leq U^{AF_{\bar{\boldsymbol{x}}}}_{min} + hK_2. \tag{5.6}$$

(ii)

$$U^{AF_{\bar{\boldsymbol{x}}}}_{min} :\leq U^{AF^{\bar{\boldsymbol{x}}}}_{min}. \tag{5.7}$$

(iii) If $\boldsymbol{x_i}$ is Accepted before $\boldsymbol{x_j}$ then $U^{AF_{\boldsymbol{x_i}}}_{min} :\leq U^{AF_{\boldsymbol{x_j}}}_{min}$.

(iv) If $U_{max}^{AF_{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + hK_2$ *then* $U_{max}^{AF^{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}} + hK_2$.

Proof :

(i)

Let $\boldsymbol{x_1}$ be a mesh point on $AF_{\bar{\boldsymbol{x}}}$ such that $U(\boldsymbol{x_1}) = U_{min}^{AF_{\bar{\boldsymbol{x}}}}$. Since it is on *AcceptedFront* immediately before $\bar{\boldsymbol{x}}$ is *Accepted*, there exists at that time a *Considered* mesh point $\boldsymbol{x_2}$ adjacent to $\boldsymbol{x_1}$. Thus, $V(\boldsymbol{x_2}) \leq U(\boldsymbol{x_1}) + K(\boldsymbol{x_2}, \frac{\boldsymbol{x_2}-\boldsymbol{x_1}}{\|\boldsymbol{x_2}-\boldsymbol{x_1}\|})\|\boldsymbol{x_2} - \boldsymbol{x_1}\|$. Since $\bar{\boldsymbol{x}}$ is about to be *Accepted*, $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) \leq V(\boldsymbol{x_2}) \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + hK_2$. On the other hand, $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) = U(\tilde{\boldsymbol{x}}) + K(\bar{\boldsymbol{x}}, \frac{\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\|\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}\|})\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|$ for some $\tilde{\boldsymbol{x}} \in AF_{\bar{\boldsymbol{x}}}$. Thus, $U(\bar{\boldsymbol{x}}) \geq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + h_{min}K_1$.

(ii)

As $\bar{\boldsymbol{x}}$ is *Accepted*, several mesh points might be removed from the *AcceptedFront*, but the only point possibly added to the *AcceptedFront* is $\bar{\boldsymbol{x}}$ itself. ($\bar{\boldsymbol{x}}$ will be added if there still is a not-yet-*Accepted* mesh point adjacent to it.) Since $U_{min}^{AF_{\bar{\boldsymbol{x}}}} \leq U(\bar{\boldsymbol{x}})$, it follows that $U_{min}^{AF_{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}}$.

(iii)

Trivially follows by induction from the inequality 5.7.

(iv)

Since $\bar{\boldsymbol{x}}$ is the only point possibly added to the *AcceptedFront*,

$$U_{max}^{AF^{\bar{\boldsymbol{x}}}} \leq \max\left(U_{max}^{AF_{\bar{\boldsymbol{x}}}}, U(\bar{\boldsymbol{x}})\right) \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + hK_2 \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}} + hK_2.$$

□

Remark 5.1.5. It immediately follows from the above Lemma that if $q_2 \leq q_1 + hK_2$ then $U_{max}^{AF} \leq U_{min}^{AF} + hK_2$ at all times. Thus, if the exit cost q is approximately constant on $\partial\Omega$ then the AF will be approximately a level set of U throughout the computation. Moreover, even if q is not approximately constant, the AF will still approximate a level set of U as soon as U_{min}^{AF} becomes bigger than $(q_2 - hK_2)$.

5.1.4 Uniform Lipschitz-continuity.

Lemma 5.1.6 (Uniform Lipschitz-continuity).

(i) If $\boldsymbol{x_1}$ *and* $\boldsymbol{x_2}$ *are two adjacent mesh points inside* Ω *then*

$$|U(\boldsymbol{x_1}) - U(\boldsymbol{x_2})| \leq L_1\|\boldsymbol{x_1} - \boldsymbol{x_2}\|, \tag{5.8}$$

where $L_1 = \eta K_2$.

(ii) Let $L_2 = \eta L_1$. *If* $\nabla U(\boldsymbol{x})$ *is defined for some* $\boldsymbol{x} \in \Omega \backslash \partial\Omega$ *such that* $d(\boldsymbol{x}) > h$ *(i.e.,* $\boldsymbol{x}$ *is not in a simplex immediately adjacent to* $\partial\Omega$*) then*

$$\|\nabla U(\boldsymbol{x})\| \leq L_2. \tag{5.9}$$

(iii) Finally, for arbitrary points $\boldsymbol{x_1}, \boldsymbol{x_2} \in \Omega$,

$$|U(\boldsymbol{x_1}) - U(\boldsymbol{x_2})| \leq L_2 \|\boldsymbol{x_1} - \boldsymbol{x_2}\|. \tag{5.10}$$

Proof :

(i)

Suppose $\boldsymbol{x_1}, \boldsymbol{x_2} \in \Omega \backslash \partial\Omega$ are two adjacent mesh points. Without loss of generality, assume that $\boldsymbol{x_1}$ was *Accepted* before $\boldsymbol{x_2}$. Thus, immediately before $\boldsymbol{x_2}$ is *Accepted*, $\boldsymbol{x_1}$ will still be on the $AcceptedFront$ and

$$\begin{aligned} U(\boldsymbol{x_2}) &\leq K(\boldsymbol{x_2}, \frac{\boldsymbol{x_1} - \boldsymbol{x_2}}{\|\boldsymbol{x_1} - \boldsymbol{x_2}\|}) \|\boldsymbol{x_1} - \boldsymbol{x_2}\| + U(\boldsymbol{x_1}) \leq \\ &\leq K_2 \|\boldsymbol{x_1} - \boldsymbol{x_2}\| + U(\boldsymbol{x_1}) \leq L_1 \|\boldsymbol{x_1} - \boldsymbol{x_2}\| + U(\boldsymbol{x_1}). \end{aligned}$$

Since U's are not necessarily *Accepted* in the ascending order, it is not generally true that $U(\boldsymbol{x_2}) \geq U(\boldsymbol{x_1})$, but from Lemma 5.1.4,

$$\begin{aligned} U(\boldsymbol{x_1}) &\leq U_{min}^{AF\boldsymbol{x_1}} + hK_2 \leq U_{min}^{AF\boldsymbol{x_2}} + hK_2 \leq \\ &\leq U(\boldsymbol{x_2}) + hK_2 = U(\boldsymbol{x_2}) + \eta K_2 h_{min} \leq U(\boldsymbol{x_2}) + L_1 \|\boldsymbol{x_1} - \boldsymbol{x_2}\|, \end{aligned}$$

which concludes the proof of the inequality 5.8.

(ii)

Let $\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}$ be the vertices of the simplex in the mesh X which contains $\boldsymbol{x}$. Since $d(\boldsymbol{x}) > h$, we know that $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, and $\boldsymbol{x_3}$ are also inside Ω, i.e., not on the boundary. Inside each simplex, U is defined by the linear interpolation and ∇U is a constant. Whatever the direction of ∇U, a straight line parallel to it passes through one of the vertices and intersects the opposite side of the triangle. Without loss of generality, assume that that line passes through $\boldsymbol{x_1}$ and intersects the side $\boldsymbol{x_2 x_3}$ at the point $\boldsymbol{x_4}$. Since $\boldsymbol{x_4}$ lies on $\boldsymbol{x_2 x_3}$,

either $(\boldsymbol{x_2} - \boldsymbol{x_1}) \cdot (\boldsymbol{x_4} - \boldsymbol{x_1}) \geq \|\boldsymbol{x_4} - \boldsymbol{x_1}\|^2$ or $(\boldsymbol{x_3} - \boldsymbol{x_1}) \cdot (\boldsymbol{x_4} - \boldsymbol{x_1}) \geq \|\boldsymbol{x_4} - \boldsymbol{x_1}\|^2$. Without loss of generality, assume the latter. Since $\|\boldsymbol{x_4} - \boldsymbol{x_1}\| \geq h_{min}$,

$$\|\nabla U\| h_{min} \;\leq\; \|\nabla U\| \|\boldsymbol{x_4} - \boldsymbol{x_1}\| \;=\; |u(\boldsymbol{x_4}) - u(\boldsymbol{x_1})| \;\leq\; |u(\boldsymbol{x_3}) - u(\boldsymbol{x_1})| \;\leq\; L_1 h.$$

Thus,

$$\|\nabla U\| \;\leq\; L_1 \frac{h}{h_{min}} \;=\; L_2.$$

(iii)

Obvious, since U is piecewise linear, with the slope bounded by L_2 in every simplex. □

Remark 5.1.7. These results seem somewhat disappointing since the true value function u has the Lipschitz constant $L \leq K_2$. A better estimate for L_1 can be obtained for the small enough h's, utilizing the smoothness of the running cost function K. A better estimate for L_2 can be obtained for the small enough h's if the exit cost function q is sufficiently smooth. However, to prove the uniform convergence of U to the value function u, it is necessary to show that some such L_2 independent of h does indeed exist. The dependence of L_2 upon η is not dangerous: if the triangulated mesh X_r does not become more and more "degenerate" as $h_r \to 0$, then η_r will be bounded.

"The fact that it works is immaterial..."

A verdict overheard after a joint Math/Applied Math Colloquium

5.2 Convergence to viscosity solution.

Without further ado,[2]...

Theorem 5.2.1 (Convergence of numerical method). *Consider a sequence of meshes* $\{X_r\}$ *such that* $h_r \to 0$, *but* $\eta^r = \frac{h_r}{h_{r_{min}}} < \eta$ *as* $r \to \infty$. *Let* U^r *be the approximate solution obtained on the mesh* X_r *by the method described in Chapter 4.*

Claim: *As* $h_r \to 0$ $U^r \to u$ *uniformly, where* u *is the viscosity solution of the equation (2.15).*

Recall that, according to the traditional definition (see [15]), a bounded, uniformly continuous function u is the viscosity solution of equation (2.15) if the following holds for each $\phi \in C_c^\infty(\Omega)$:

(i)

if $u - \phi$ has a local minimum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla \phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} \leq 0; \tag{5.11}$$

(ii)

if $u - \phi$ has a local maximum at $\boldsymbol{x_0} \in \Omega$ then

$$\min_{\boldsymbol{a} \in S_1} \{\nabla \phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} \geq 0; \tag{5.12}$$

Proof. Since $\{U^r\}$ are bounded and uniformly Lipschitz-continuous, by the Arzela–Ascoli theorem, there exists a subsequence $\{X_p\}$ of the sequence $\{X_r\}$ such that $h_p \to 0$ as $p \to \infty$ and a function u such that $U^p \to u$ uniformly as $p \to \infty$. Boundedness and uniform continuity of u immediately follow from the properties of U^p.

(i)

Consider any function $\phi \in C_c^\infty(\Omega)$ and suppose that $(u - \phi)$ has a strict local minimum at $\boldsymbol{x_0} \in \Omega$. Define B_δ to be the closed ball of radius δ around $\boldsymbol{x_0}$. Then there exists some $\delta > 0$ such that $B_\delta \subset \Omega$ and $\boldsymbol{x} \in B_\delta$ implies

[2] I would like to thank L.C.Evans and J.A.Sethian for their comments and suggestions regarding the general structure of this proof.

$$(u-\phi)(\boldsymbol{x_0}) < (u-\phi)(\boldsymbol{x}). \tag{5.13}$$

If $D_2(\boldsymbol{x})$ is the matrix of second derivatives of $\phi(\boldsymbol{x})$ then $\exists \mu > 0$ such that $\|D_2(\boldsymbol{x})\|_2 \leq \mu$ for all $\boldsymbol{x} \in B_\delta$. Let now ${\boldsymbol{x_0}}^p$ be a minimum point for $(U^p - \phi)$ over B_δ; from (5.13) and from the uniform convergence of U^p's it follows that

$$\lim_{p\to\infty} {\boldsymbol{x_0}}^p = \boldsymbol{x_0}. \tag{5.14}$$

If ${\boldsymbol{x_0}}^p$ is not a grid point of X_p, there exists a unique simplex s containing ${\boldsymbol{x_0}}^p$. Define ${\boldsymbol{x_1}}^p$ to be the vertex of s closest to ${\boldsymbol{x_0}}^p$. (We define ${\boldsymbol{x_1}}^p = {\boldsymbol{x_0}}^p$ if ${\boldsymbol{x_0}}^p$ is a mesh point.) Since ϕ is smooth and U^p is linear on s, we know that $\nabla U^p({\boldsymbol{x_0}}^p) = \nabla\phi({\boldsymbol{x_0}}^p)$ and

$$(U^p - \phi)({\boldsymbol{x_1}}^p) \leq (U^p - \phi)({\boldsymbol{x_0}}^p) + \frac{\mu {h_p}^2}{2}. \tag{5.15}$$

Moreover, using the inequality 5.15, we obtain for every $\boldsymbol{x} \in B_\delta$,

$$\begin{aligned}\phi(\boldsymbol{x}) - \phi({\boldsymbol{x_1}}^p) &= (\phi(\boldsymbol{x}) - \phi({\boldsymbol{x_0}}^p)) + (\phi({\boldsymbol{x_0}}^p) - \phi({\boldsymbol{x_1}}^p)) \leq \\ &\leq (U^p(\boldsymbol{x}) - U^p({\boldsymbol{x_0}}^p)) + (U^p({\boldsymbol{x_0}}^p) - U^p({\boldsymbol{x_1}}^p) + \frac{\mu {h_p}^2}{2}) = \\ &= U^p(\boldsymbol{x}) - U^p({\boldsymbol{x_1}}^p) + \frac{\mu {h_p}^2}{2}. \end{aligned} \tag{5.16}$$

Since $\|{\boldsymbol{x_0}}^p - {\boldsymbol{x_1}}^p\| \leq h_p$, it is also clear that $\lim_{p\to\infty} {\boldsymbol{x_1}}^p = \boldsymbol{x_0}$. So, for big enough p, $h_p \frac{K_2}{K_1} \leq \delta$; thus, by the update formula 4.7, $\exists \tilde{\boldsymbol{x}}^{\boldsymbol{p}} \in AF_{{\boldsymbol{x_1}}^p} \bigcap B_\delta$ such that

$$U^p({\boldsymbol{x_1}}^p) = \tau_p K({\boldsymbol{x_1}}^p, \boldsymbol{a}^p) + U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}), \tag{5.17}$$

where $\tau_p = \|\tilde{\boldsymbol{x}}^p - {\boldsymbol{x_1}}^p\|$ and $\boldsymbol{a}^p = \frac{\tilde{\boldsymbol{x}}^p - {\boldsymbol{x_1}}^p}{\|\tilde{\boldsymbol{x}}^p - {\boldsymbol{x_1}}^p\|}$.
Using the smoothness of ϕ, the inequality 5.16, and the equality 5.17, we obtain

$$\begin{aligned}\nabla\phi({\boldsymbol{x_1}}^p)\cdot\boldsymbol{a}^p + K({\boldsymbol{x_1}}^p, \boldsymbol{a}^p) &\leq \frac{\phi({\boldsymbol{x_1}}^p + \tau_p\boldsymbol{a}^p) - \phi({\boldsymbol{x_1}}^p)}{\tau_p} + K({\boldsymbol{x_1}}^p, \boldsymbol{a}^p) + \tau_p\mu \leq \\ &\leq \frac{U^p({\boldsymbol{x_1}}^p + \tau_p\boldsymbol{a}^p) - U^p({\boldsymbol{x_1}}^p) + \frac{\mu {h_p}^2}{2}}{\tau_p} + K({\boldsymbol{x_1}}^p, \boldsymbol{a}^p) + \tau_p\mu = \\ &= \frac{U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}) - U^p({\boldsymbol{x_1}}^p) + \tau_p K({\boldsymbol{x_1}}^p, \boldsymbol{a}^p)}{\tau_p} + \frac{\mu {h_p}^2}{2\tau_p} + \tau_p\mu = \\ &= \frac{\mu {h_p}^2}{2\tau_p} + \tau_p\mu. \end{aligned} \tag{5.18}$$

Since $\tilde{\boldsymbol{x}}$ lies on the $AF_{\boldsymbol{x_1}^p}$ and $\tau_p = \|\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p\|$, it is at least as big as the minimal triangle height in the mesh X_p, i.e., $\tau_p \geq \frac{h_p}{\eta}$. On the other hand, $\tau_p \leq h_p \frac{K_2}{K_1}$ because $\tilde{\boldsymbol{x}}^p \in \mathrm{NF}(\boldsymbol{x_1}^p)$. Combining these bounds with inequality 5.18, we see that

$$\nabla\phi(\boldsymbol{x_1}^p) \cdot \boldsymbol{a}^p + K(\boldsymbol{x_1}^p, \boldsymbol{a}^p) \leq \mu(\frac{\eta}{2} + \frac{K_2}{K_1})h_p. \tag{5.19}$$

The sequence $\{\boldsymbol{a}^p\}$ has to have a subsequence converging to some vector $\boldsymbol{b} \in S_1$; we restrict our attention to that subsequence, but will still use the subscript p to avoid further cluttering of the notation. Now (finally!) we can use the continuity of K, smoothness of ϕ and the uniformity of convergence of U^p to pass to a limit as $p \to \infty$ in the inequality 5.19:

$$\nabla\phi(\boldsymbol{x_0}) \cdot \boldsymbol{b} + K(\boldsymbol{x_0}, \boldsymbol{b}) \leq 0,$$

which completes the first half of the proof, since

$$\min_{\boldsymbol{a} \in S_1} \{\nabla\phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} \leq \nabla\phi(\boldsymbol{x_0}) \cdot \boldsymbol{b} + K(\boldsymbol{x_0}, \boldsymbol{b}).$$

(ii)

Consider any function $\phi \in C_c^\infty(\Omega)$ and suppose that $(u - \phi)$ has a strict local maximum at $\boldsymbol{x_0} \in \Omega$. Define B_δ to be the closed ball of radius δ around $\boldsymbol{x_0}$. Then there exists some $\delta > 0$ such that $B_\delta \subset \Omega$ and $\boldsymbol{x} \in B_\delta$ implies

$$(u - \phi)(\boldsymbol{x_0}) > (u - \phi)(\boldsymbol{x}). \tag{5.20}$$

If $\nabla\phi(\boldsymbol{x_0}) = 0$ then the inequality 5.12 is trivially satisfied. Thus, we will further assume that $\|\nabla\phi(x)\| \geq \nu > 0$ for all $\boldsymbol{x} \in B_\delta$. If $D_2(\boldsymbol{x})$ is the matrix of second derivatives of $\phi(\boldsymbol{x})$ then $\exists\mu > 0$ such that $\|D_2(\boldsymbol{x})\|_2 \leq \mu$ for all $\boldsymbol{x} \in B_\delta$. Let now $\boldsymbol{x_0}^p$ be a maximum point for $(U^p - \phi)$ over B_δ; from (5.20) and from the uniform convergence of U^p's it follows that

$$\lim_{p \to \infty} \boldsymbol{x_0}^p = \boldsymbol{x_0}. \tag{5.21}$$

If $\boldsymbol{x_0}^p$ is not a grid point of X_p, there exists a unique simplex s containing $\boldsymbol{x_0}^p$. Define $\boldsymbol{x_1}^p$ to be the vertex of s closest to $\boldsymbol{x_0}^p$. (We define $\boldsymbol{x_1}^p = \boldsymbol{x_0}^p$ if $\boldsymbol{x_0}^p$ is a mesh point.) Since ϕ is smooth and U^p is linear on s, we know that $\nabla U^p(\boldsymbol{x_0}^p) = \nabla\phi(\boldsymbol{x_0}^p)$ and

$$(U^p - \phi)(\boldsymbol{x_1}^p) \geq (U^p - \phi)(\boldsymbol{x_0}^p) - \frac{\mu {h_p}^2}{2}. \tag{5.22}$$

Moreover, using the inequality 5.22, we obtain for every $\boldsymbol{x} \in B_\delta$,

$$\begin{aligned}\phi(\boldsymbol{x}) - \phi(\boldsymbol{x_1}^p) = (\phi(\boldsymbol{x}) - \phi(\boldsymbol{x_0}^p)) + (\phi(\boldsymbol{x_0}^p) - \phi(\boldsymbol{x_1}^p)) \geq \\ \geq (U^p(\boldsymbol{x}) - U^p(\boldsymbol{x_0}^p)) + (U^p(\boldsymbol{x_0}^p) - U^p(\boldsymbol{x_1}^p) - \frac{\mu {h_p}^2}{2}) = \\ = U^p(\boldsymbol{x}) - U^p(\boldsymbol{x_1}^p) - \frac{\mu {h_p}^2}{2}. \end{aligned} \tag{5.23}$$

Since $\|\boldsymbol{x_0}^p - \boldsymbol{x_1}^p\| \leq h_p$, it is also clear that $\lim_{p\to\infty} \boldsymbol{x_1}^p = \boldsymbol{x_0}$.

If you board the wrong train, it is no use running along the corridor in the other direction.

DIETRICH BONHOEFFER (1906 - 1945) GERMAN THEOLOGIAN

As proven in section 2.2.6,

$$\min_{\boldsymbol{a}\in S_1} \{\nabla\phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\} = \min_{\boldsymbol{a}\in S_1^{\phi,\boldsymbol{x_0}}} \{\nabla\phi(\boldsymbol{x_0}) \cdot \boldsymbol{a} + K(\boldsymbol{x_0}, \boldsymbol{a})\}. \tag{5.24}$$

Thus, to prove (5.12), we only need to consider $\boldsymbol{a} \in S_1^{\phi,\boldsymbol{x_0}}$, i.e., only $\boldsymbol{a}$ such that

$$\boldsymbol{a} \cdot \nabla\phi(\boldsymbol{x_0}) \leq -\frac{K_1}{K_2}\|\nabla\phi(\boldsymbol{x_0})\| \leq -\nu\frac{K_1}{K_2}. \tag{5.25}$$

Suppose a particular $\boldsymbol{a} \in S_1^{\phi,\boldsymbol{x_0}}$ was chosen. We would like to show that, for sufficiently small h_p,

if we start at $\boldsymbol{x_1}^p$ and go some distance $\tau_p = O(h_p)$ in the direction $\boldsymbol{a}$, then we will have to intersect the $AF_{\boldsymbol{x_1}^p}$ **(*)**

If the local maximum were attained at the mesh point (i.e., the case $\boldsymbol{x_1}^p = \boldsymbol{x_0}^p$) and the test function ϕ were linear, then **(*)** would be almost obvious: ϕ would be linearly decreasing in the direction $\boldsymbol{a}$, so would be U^p because of the local maximum condition, and, as we know from Lemma 5.1.4,

$$U^p(x) \geq U_{min}^{AF_{\boldsymbol{x_1}^p}} + \frac{h_p}{\eta}K_1$$

for every mesh point $x \in X_p$ *Accepted* after the point $\boldsymbol{x_1}^p$. Since ϕ is generally not linear and $\boldsymbol{x_1}^p \neq \boldsymbol{x_0}^p$, we will have to be more careful.

Suppose we start moving from $\boldsymbol{x_1}^p$ in the direction $\boldsymbol{a}$ for some time t_p. Using the inequality 5.23, the smoothness of ϕ, and the inequality 5.25, we obtain

$$U^p(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) - U^p(\boldsymbol{x_1}^p) \leq (\phi(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) - \phi(\boldsymbol{x_1}^p)) + \frac{\mu {h_p}^2}{2} \leq$$
$$\leq (t_p(\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}) + \frac{\mu {t_p}^2}{2}) + \frac{\mu {h_p}^2}{2} \leq -\nu t_p\frac{K_1}{K_2} + \mu\frac{{h_p}^2 + {t_p}^2}{2}. \tag{5.26}$$

In order to prove **(*)** we will need the following inequality to be satisfied:

$$-\nu t_p\frac{K_1}{K_2} + \mu\frac{{h_p}^2 + {t_p}^2}{2} \leq -h_pK_2. \tag{5.27}$$

Let $t_p = Ah_p$. We will now show that the constant A can be chosen such that (5.27) holds for small enough h_p. Indeed, (5.27) can be rewritten as

$$h_p^2\left(\mu\frac{1+A^2}{2}\right) + h_p\left(K_2 - \nu A\frac{K_1}{K_2}\right) \leq 0 \tag{5.28}$$

If A is such that $\left(K_2 - \nu A\frac{K_1}{K_2}\right) > 0$, then (5.27) is satisfied for all the

$$h_p \in \left[0, \left(\nu A\frac{K_1}{K_2} - K_2\right)\frac{2}{\mu(1+A^2)}\right].$$

Thus, choosing any $A > \left(\frac{K_2^2}{\nu K_1}\right)$, we ensure that (5.27) is satisfied for the sufficiently small h_p. Combining this with the inequality 5.26, and using the monotonicity result in Lemma 5.1.4, we see that

$$U^p(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) \leq U^p(\boldsymbol{x_1}^p) - h_pK_2 \leq U_{min}^{AF_{\boldsymbol{x_1}^p}},$$

i.e., the point $(\boldsymbol{x_1}^p + t_p\boldsymbol{a})$ cannot be inside the $AF_{\boldsymbol{x_1}^p}$. Since $\boldsymbol{x_1}^p$ is inside $AF_{\boldsymbol{x_1}^p}$, that means that **(*)** holds: there exists some $\tau_p \in [0, t_p]$ such that

$$\tilde{\boldsymbol{x}}^{\boldsymbol{p}} = (\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}) \in AF_{\boldsymbol{x_1}^p}.$$

By the Lemma 5.1.1,

$$U^p(\boldsymbol{x_1}^p) = W^p(\boldsymbol{x_1}^p) \leq \tau_pK(\boldsymbol{x_1}^p, \boldsymbol{a}) + U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}), \tag{5.29}$$

The remainder of the proof is similar to what we have done to prove **(i)**.
Using the smoothness of ϕ, the inequality 5.23, and the inequality 5.29, we obtain

$$\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}+K(\boldsymbol{x_1}^p,\boldsymbol{a})\ \geq\ \frac{\phi(\boldsymbol{x_1}^p+\tau_p\boldsymbol{a})-\phi(\boldsymbol{x_1}^p)}{\tau_p}+K(\boldsymbol{x_1}^p,\boldsymbol{a})-\tau_p\mu\ \geq$$
$$\geq\frac{U^p(\boldsymbol{x_1}^p+\tau_p\boldsymbol{a})-U^p(\boldsymbol{x_1}^p)-\frac{\mu{h_p}^2}{2}}{\tau_p}+K(\boldsymbol{x_1}^p,\boldsymbol{a})-\tau_p\mu=$$
$$=\frac{U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}})-U^p(\boldsymbol{x_1}^p)+\tau_pK(\boldsymbol{x_1}^p,\boldsymbol{a})}{\tau_p}-\frac{\mu{h_p}^2}{2\tau_p}-\tau_p\mu\geq$$
$$\geq-\frac{\mu{h_p}^2}{2\tau_p}-\tau_p\mu. \qquad (5.30)$$

Since $\tilde{\boldsymbol{x}}$ lies on the $AF_{\boldsymbol{x_1}^p}$ and $\tau_p=\|\tilde{\boldsymbol{x}}^p-\boldsymbol{x_1}^p\|$, it is at least as big as the minimal triangle height in the mesh X_p, i.e., $\tau_p\geq\frac{h_p}{\eta}$. On the other hand, $\tau_p\leq t_p=Ah_p$, where A is a constant chosen for this particular ϕ, but independent of h_p. Combining these bounds with inequality 5.30, we see that

$$\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}+K(\boldsymbol{x_1}^p,\boldsymbol{a})\geq-\mu(\frac{\eta}{2}+A)h_p. \qquad (5.31)$$

We can now use the continuity of K, smoothness of ϕ and the uniformness of convergence of U^p to pass to a limit as $p\to\infty$ in the inequality 5.31:

$$\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a}+K(\boldsymbol{x_0},\boldsymbol{a})\geq 0,$$

which completes the proof of inequality 5.12, since $\boldsymbol{a}$ was chosen to be an arbitrary vector in $S_1^{\phi,\boldsymbol{x_0}}$.

In this proof we have several times passed to a subsequence. Suppose some other subsequence of U^r converges to a different limit $\bar{u}$. The above argument could be repeated for that subsequence to prove that $\bar{u}$ also satisfies (5.12) and (5.11). The uniqueness of the viscosity solution (proven in [16], [15]) implies $u=\bar{u}$; thus, the entire sequence U^r converges to u uniformly as $r\to\infty$. □

Remark 5.2.2. Unfortunately, the above proof demonstrates only the fact of convergence, yielding no rate of convergence estimates. Another approach, used as a motivation in section 4.3, would permit to compute error bounds, but, to the best of our knowledge, it would also require some additional assumptions about the regularity of ϵ-suboptimal controls.

Remark 5.2.3. This proof uses the continuity of the running cost function K, but not the Lipschitz-continuity. Thus, if the viscosity solution can be defined for $K\in C(\Omega)$ then the

above proof will still be valid. The majority of the control-theoretic papers, which we are referring to in this work, require the running cost K (or, equivalently, the speed of motion f) to be Lipschitz-continuous. Nevertheless, the value function u can be defined for a much broader class of control problems, including those, for which K is discontinuous. Some examples of our method applied to such problems can be found in Chapter 8. Even though we do not have a proof of convergence, the numerical evidence confirms that the method described above works correctly even in that general case. This is not surprising. After all, as noted in section 2.2, Bellman's optimality principle is valid even when running cost is very ill-behaved, and our numerical methods merely mimic the logic of that principle.

Remark 5.2.4. The generally accepted definition of the viscosity solution (see [16], for example) uses the test functions $\phi \in C^1(\Omega)$. However, as shown in [15], the definition using the test functions $\phi \in C_c^\infty(\Omega)$ is equivalent. This enabled us to use the upper bounds on the second derivatives of ϕ in the above proof.

> God forbid that Truth should be confined to Mathematical Demonstration!
>
> —BLAKE

NB: This proof of convergence, as well as the preceding lemmas, can be easily repeated for the corresponding method in higher dimensions. Moreover, it is our not-so-secret belief that the following general conjecture is true:
If the update formula 4.7 in the description of the method is replaced by any other update formula such that

- *i the update formula is consistent (converges to the PDE as $h \to 0$),*
- *ii the update formula is upwinding (the update is computed/accepted only from the simplex, which contains the characteristic direction),*
- *iii the update formula is stable (there exists a uniform bound for U),*

then the resulting numerical solutions will converge to the viscosity solution of Hamilton-Jacobi-Bellman PDE.

This conjecture is the basis for the hybrid methods discussed in section 6.2.

Chapter 6

Hybrid Methods for Anisotropic Problems

In Chapter 4 we introduced a new single-pass method for solving the Hamilton-Jacobi-Bellman PDE 2.15 corresponding to the anisotropic optimal trajectory problems described in section 2.2. We will now explore the correspondence between these problems and a class of the anisotropic front expansion (contraction) problems described in section 3.2.
Our goal in this exploration is twofold:

- to determine a set of the anisotropic front expansion (contraction) problems, which can be solved efficiently by the Ordered Upwinding Method introduced in Chapter 4;
- to use this correspondence to construct a family of hybrid Ordered Upwinding Methods utilizing the finite difference operators described in section 3.4.

Pereant qui ante nos nostra dixerunt.
"To the devil with those who published before us."

Aelius Donatus (4th Century)
Quoted by St. Jerome, his pupil

The phenomenon of anisotropy is one of importance in many application domains. After deriving the formulae in section 6.1, we have found that many of these issues had already been the focus of attention in geometric optics [33], geophysics [37, 17], tomography

[26], and crystal growth [52, 49]. Thus, we are unsure if any of the results in section 6.1 are original.

We will try to point out the connections to the terminology and results in prior publications whenever possible[1]. Our goal is to emphasize the connections in an application-neutral manner, concentrating on the properties of the particular class of static Hamilton-Jacobi PDEs.

6.1 Front propagation vs. control theory: the direct mapping

Recall from section 3.2 that if the front is only expanding (or contracting), its evolution can be fully described by the PDE :

$$\|\nabla u(\boldsymbol{x})\| F\left(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}\right) = 1,$$

$$u = 0 \text{ on } \Gamma_0,$$

where F is the speed of the front in the direction normal to itself, Γ_0 is the initial position of the front, and $u(\boldsymbol{x})$ is the time when the front passes through the point $\boldsymbol{x}$.

If F depends only on $\boldsymbol{x}$, the above PDE reduces to the Eikonal equation, which (as shown in section 2.2.4) can be interpreted from the control-theoretic perspective as an equation describing the isotropic optimal trajectory problem.

Our goal for this section is to derive a corresponding interpretation for a wider class of front propagation problems. We recall from section 2.2.5 that the min-time optimal trajectory problem is described by the Hamilton-Jacobi-Bellman PDE 2.24 :

$$\max_{\boldsymbol{a} \in S_1} \{(\nabla u(\boldsymbol{x}) \cdot (-\boldsymbol{a})) f(\boldsymbol{x}, \boldsymbol{a})\} = 1.$$

The Hamiltonian in the above equation is convex in ∇u. Thus, it is clear that the control-theoretic interpretation of the PDE 3.2 is possible only when the Hamiltonian $H(\nabla u, \boldsymbol{x}) = \|\nabla u\| F\left(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|}\right)$ is also convex in the first argument[2]. For these two problems to be

[1] Incidentally, the above quote from Aelius Donatus is also often attributed to Abinitus Provenius (1st Century B.C.).

[2] If the Hamiltonian is not convex, the front expansion problem can be restated in terms of the differential games theory [20, 5]; we are currently investigating the applicability of our single-pass methods to these more general problems.

equivalent, the Hamiltonians have to be equal:

$$\|\nabla u(\boldsymbol{x})\| F\left(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}\right) = \max_{\boldsymbol{a}\in S_1}\{(\nabla u(\boldsymbol{x})\cdot(-\boldsymbol{a}))f(\boldsymbol{x},\boldsymbol{a})\},$$

or, equivalently, functions F and f have to be such that

$$F(\boldsymbol{x},\boldsymbol{n}) = \max_{\boldsymbol{a}\in S_1}\{(\boldsymbol{n}\cdot(-\boldsymbol{a}))f(\boldsymbol{x},\boldsymbol{a})\}. \tag{6.1}$$

This immediately yields the relationship between the speed functions F and f discussed in the next section.

> Common sense and a sense of humor are the same thing, moving at different speeds. A sense of humor is just common sense, dancing.
>
> CLIVE JAMES, Australian writer

Remark 6.1.1. One should be careful to distinguish these two speed functions: $F(\boldsymbol{x},\boldsymbol{n})$ is the speed of the front's movement in the direction normal to itself[3] ($\boldsymbol{n} = \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}$), whereas $f(\boldsymbol{x},\boldsymbol{a})$ is the speed of the vehicle's motion[4] in the direction $\boldsymbol{a}$. Correspondingly, the correct $\boldsymbol{n}$ is fully determined by the gradient direction of the function $u(\boldsymbol{x})$, whereas the optimal $\boldsymbol{a}\in S_1$ is determined by the direction of the characteristic passing through the point $\boldsymbol{x}$ and, therefore, is a function of the particular Hamilton-Jacobi-Bellman equation. (See Figure 4.2 for the illustration of the gradient and characteristic directions for an expanding ellipse.)

Remark 6.1.2. The difference between these two formulations is less apparent for the Eikonal equation since in that special case the characteristic direction is always exactly the opposite of the gradient direction (provided, of course, that the gradient is defined at the point) :

$$F(\boldsymbol{x},\boldsymbol{n}) = \max_{\boldsymbol{a}\in S_1}\{(\boldsymbol{n}\cdot(-\boldsymbol{a}))f(\boldsymbol{x})\} = f(\boldsymbol{x}) = F(\boldsymbol{x}).$$

6.1.1 The Alpha and Omega of Anisotropy

In R^2, it is more convenient to rewrite the equation 6.1 in terms of trigonometric functions. Let $\boldsymbol{a} = \begin{bmatrix}\cos(\alpha)\\ \sin(\alpha)\end{bmatrix}$ and $\boldsymbol{n} = \begin{bmatrix}\cos(\omega)\\ \sin(\omega)\end{bmatrix}$. We will further abuse the terminology

[3] In wave physics $F(\boldsymbol{x},\boldsymbol{n})$ corresponds to the *"phase velocity"* if $\boldsymbol{n}$ is the direction normal to the wavefront [17]. In crystalline variational problems $F(\boldsymbol{x},\boldsymbol{n})$ corresponds to the *"surface free energy"* if $\boldsymbol{n}$ is the direction normal to the surface [52].

[4] In wave physics $f(\boldsymbol{x},\boldsymbol{a})$ corresponds to the *"group velocity"*, i.e., the speed, with which a blob of energy is moving in the direction $\boldsymbol{a}$ [17].

and define $F(\boldsymbol{x}, \omega) = F(\boldsymbol{x}, \boldsymbol{n})$ and $f(\boldsymbol{x}, \alpha) = f(\boldsymbol{x}, \boldsymbol{a})$. The condition for the equivalence of the front expansion and optimal trajectory problems can be rewritten as

$$F(\boldsymbol{x}, \omega) = \max_{\alpha \in [0, 2\pi]} \{\cos(\omega - (\alpha + \pi)) f(\boldsymbol{x}, \alpha)\}, \tag{6.2}$$

Remark 6.1.3. This formula already shows how to cast any min-time optimal trajectory problem as a front expansion problem. However, we are interested in finding the inverse mapping as well.

In order to obtain the geometric interpretation of the relationship between f and F, we let $\beta = \alpha + \pi$ and $g(\boldsymbol{x}, \beta) = f(\boldsymbol{x}, \beta - \pi) = f(\boldsymbol{x}, \alpha)$ and rewrite Eqn 6.2 as:

$$F(\boldsymbol{x}, \omega) = \max_{\beta \in [0, 2\pi]} \{\cos(\omega - \beta) g(\boldsymbol{x}, \beta)\}. \tag{6.3}$$

Now consider the vehicle's speed profile[5] - the polar coordinates plots $\rho = f(\boldsymbol{x}, \theta)$ and $\rho = g(\boldsymbol{x}, \theta)$. The later is a flipped (center symmetry applied) version of the former; see Figure 6.1.

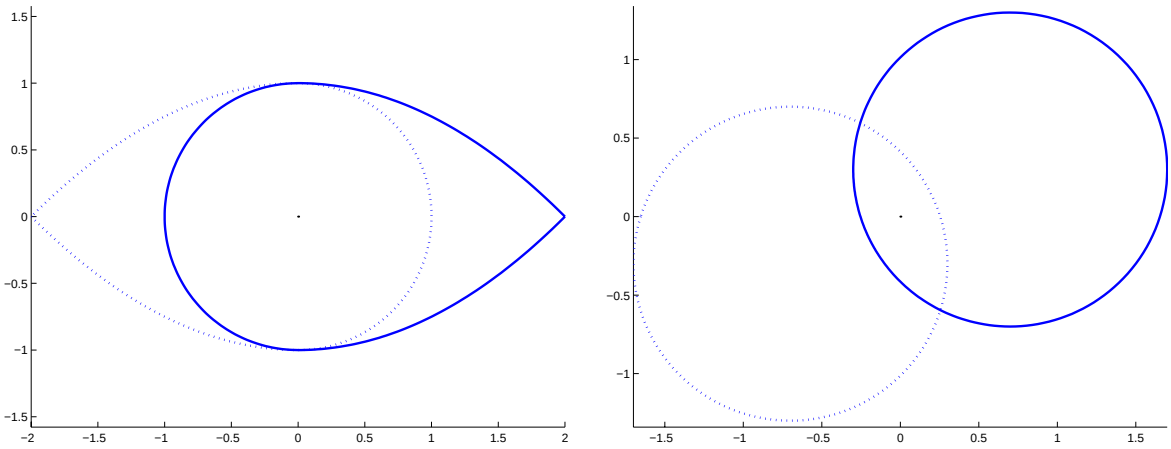


Figure 6.1: Two examples of vehicle's speed profiles $\rho = f(\theta)$. The dotted line shows the "flipped" version of the profile, i.e., $\rho = g(\theta) = f(\theta + \pi)$.

The formula 6.3 means that $F(\boldsymbol{x}, \omega)$ can be obtained by projecting the speed profile $\rho = g(\boldsymbol{x}, \theta)$ onto the line parallel to the vector $\boldsymbol{n}$, and then by taking the maximum

[5] In wave physics this speed profile is often referred to as a *"group-velocity curve"*, *"ray surface"*, or *"impulse-response surface"* [17].
The corresponding object in crystalline variational problems is the *"Wulff shape"* - the shape, which minimizes the free surface energy for a fixed volume with no additional constraints [52].

of this (signed) projection. Correspondingly, the optimal β is the one, which maximizes the projection onto $\boldsymbol{n}$ of the vector $(-g(\boldsymbol{x},\beta)\boldsymbol{a})$.

Thus, if the vehicle's speed profile is *strictly* convex then there exists precisely one optimal β for each ω; moreover, each β is optimal for some ω, i.e., the $\beta \mapsto \omega$ correspondence is one-to-one and onto. It is easy to show that if $\frac{\partial}{\partial\beta}g(\boldsymbol{x},\beta)$ exists and $\omega(\beta)$ is the angle for which β is optimal, then the tangent to the vehicle's speed profile at the point $(-g(\boldsymbol{x},\beta)\boldsymbol{a})$ is perpendicular to the vector $\boldsymbol{n} = \begin{bmatrix} \cos(\omega(\beta)) \\ \sin(\omega(\beta)) \end{bmatrix}$ and the distance from that tangent line to the origin is $F(\boldsymbol{x},\omega(\beta))$. This provides a geometric recipe for building the front propagation speed profile $\rho = F(\boldsymbol{x},\theta)$ using the vehicle's speed profile (see Figure 6.2)[6].

> Whoever proves his point and demonstrates the prime truth geometrically should be believed by all the world, for there we are captured.
>
> ALBRECHT DÜRER (1471-1528)

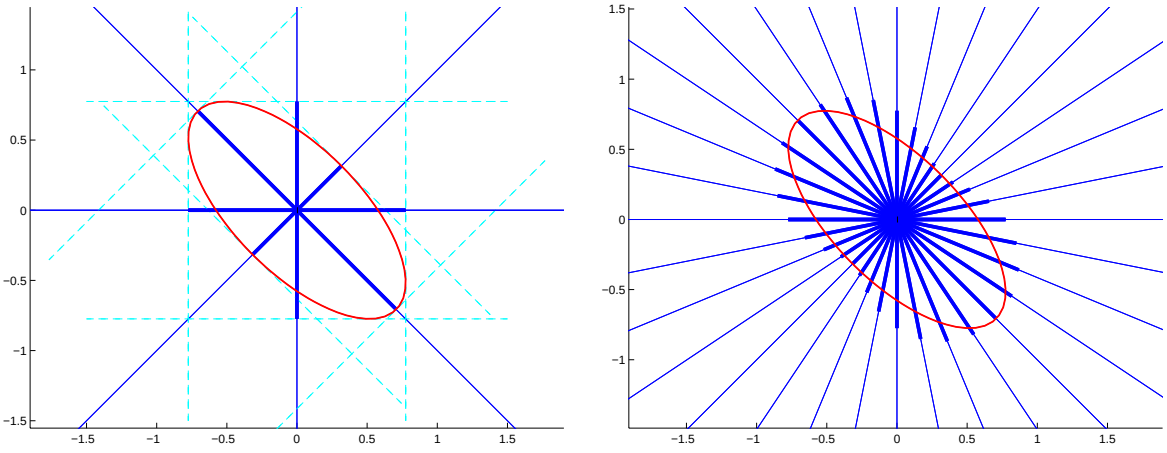


Figure 6.2: Using the vehicle motion speed profile to construct the the front propagation speed profile.

Under these assumptions we can also derive an explicit formula for $\omega(\beta)$: since β is optimal for that ω, $\frac{\partial}{\partial\beta}(\cos(\omega-\beta)g(\boldsymbol{x},\beta)) = 0$, which yields

$$\omega(\beta) = \beta - \arctan\left(\frac{\frac{\partial}{\partial\beta}g(\boldsymbol{x},\beta)}{g(\boldsymbol{x},\beta)}\right). \tag{6.4}$$

[6]This geometric construction is very common in tomography; the formulas 6.2 and 6.4 are closely related to the *Radon transform* [26].

Since both functions F and g are positive and $F(\boldsymbol{x},\omega) = \cos(\omega - \beta(\omega))g(\boldsymbol{x},\beta(\omega)$, it is clear that $\|\omega - \beta(\omega)\| < \pi/2$. Consider any $\tilde{\omega} \neq \omega$ form the interval $(\beta(\omega) - \pi/2, \beta(\omega) + \pi/2)$. Since the relationship is one-to-one and onto, $\beta(\omega) \neq \beta(\tilde{\omega})$; thus, $F(\boldsymbol{x},\tilde{\omega}) > \cos(\tilde{\omega} - \beta(\omega))g(\boldsymbol{x},\beta(\omega)$ for all such $\tilde{\omega}$. Therefore, g can be conversely defined in terms of F:

$$f(\boldsymbol{x},\alpha) = g(\boldsymbol{x},\beta) = \min_{\omega\in(\beta-\pi/2,\beta+\pi/2)} \frac{F(\boldsymbol{x},\omega)}{\cos(\omega-\beta)}. \tag{6.5}$$

If F is smooth at the point then we can also derive an explicit formula for $\beta(\omega)$: since ω is optimal for that β, $\frac{\partial}{\partial\omega}(\frac{F(\boldsymbol{x},\omega)}{\cos(\omega-\beta)}) = 0$, which yields

$$\beta(\omega) = \omega + \arctan\left(\frac{\frac{\partial}{\partial\omega}F(\boldsymbol{x},\omega)}{F(\boldsymbol{x},\omega)}\right). \tag{6.6}$$

The geometric interpretation of Eqn 6.6 is simple. Consider the front propagation speed profile - the polar coordinates plots $\rho = F(\boldsymbol{x},\theta)$. For each point $F(\boldsymbol{x},\omega)\begin{bmatrix}\cos(\omega)\\ \sin(\omega)\end{bmatrix} = F(\boldsymbol{x},\omega)\boldsymbol{n}$ on the speed profile, draw a straight line through that point perpendicular to the vector $\boldsymbol{n}$. The envelope of these lines is the figure $\rho = g(\boldsymbol{x},\theta)$, i.e., the "flipped" vehicle's speed profile $\rho = f(\boldsymbol{x},\theta)$ (see Figure 6.3, for example). More precisely, the envelope will yield the "flipped" convex hull of the original vehicle's speed profile; thus, the optimal trajectory problems with the different vehicle's speed functions will yield the same Hamilton-Jacobi-Bellman equation provided the speed profiles have the same convex hull[7].

In general, **if** the vehicle's speed profile $\rho = f(\boldsymbol{x},\theta)$

- is smooth and strictly convex **then** the functions $\omega(\alpha)$ and $\alpha(\omega)$ are one-to-one and onto (Figure 6.4A);
- is convex, but not smooth **then** the same α will be optimal for multiple ω's (Figure 6.4B);
- is convex, but not *strictly* convex **then** multiple α's will be optimal for the same ω (Figure 6.4C);

[7] This geometric construction is very common in tomography; the formulas 6.5 and 6.6 are closely related to the *inverse Radon transform* [26].
In wave physics [33], formulas expressing the relationship between the *group speed* and the *phase speed* (similar to (6.2), (6.4), (6.5), and (6.6)) were known as early as 1837!

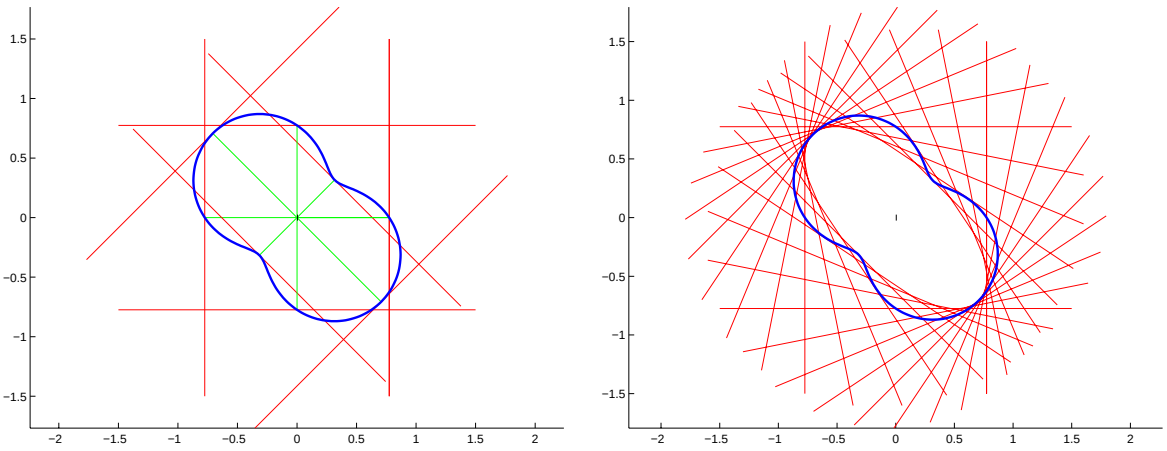


Figure 6.3: Using the front propagation speed profile to construct the the vehicle motion speed profile.

- is non-convex **then** only α's such that $f(\boldsymbol{x},\alpha)\begin{bmatrix}\cos(\alpha)\\ \sin(\alpha)\end{bmatrix}$ is on the boundary of the convex hull of the vehicle's speed profile will be optimal for some ω (Figure 6.4D).

Remark 6.1.4. Let

$$F_1(\boldsymbol{x}) = \min_{\omega} F(\boldsymbol{x},\omega), \qquad F_2(\boldsymbol{x}) = \max_{\omega} F(\boldsymbol{x},\omega),$$

and

$$f_1(\boldsymbol{x}) = \min_{\alpha} f(\boldsymbol{x},\alpha), \qquad f_2(\boldsymbol{x}) = \max_{\alpha} f(\boldsymbol{x},\alpha).$$

Then it follows from the formula 6.2 that

$$F_2(\boldsymbol{x}) = f_2(\boldsymbol{x}), \qquad \text{and } F_1(\boldsymbol{x}) \geq f_1(\boldsymbol{x}).$$

If the vehicle's speed profile $\rho = f(\boldsymbol{x},\theta)$ is convex, then it is also easy to show that $F_1(\boldsymbol{x}) = f_1(\boldsymbol{x})$ since each α is optimal for at least one ω.

6.1.2 Huygens' principle: the modified version.

The correspondence between F and f suggests an alternative application of Huygens' principle to the front propagation problem[8]:

[8] An example of similar modified Huygens' construction using Wulff shapes instead of circles to model the growth of thin films can be found in [49].

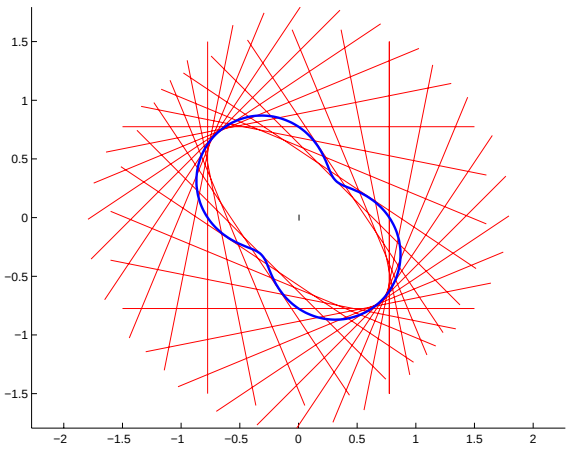


Fig.6.4A

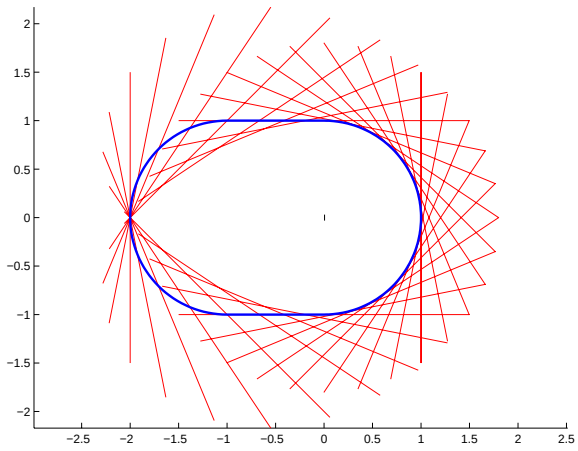


Fig.6.4B

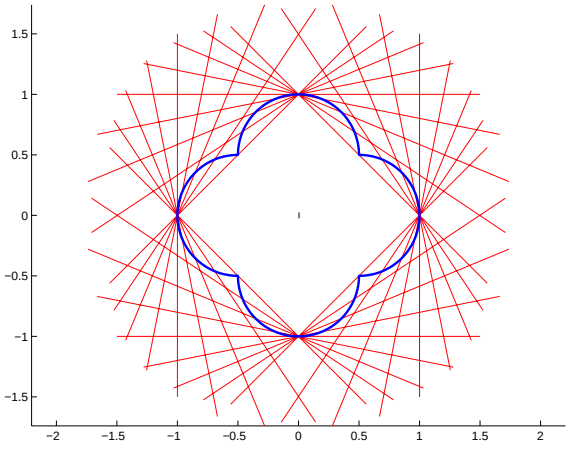


Fig.6.4C

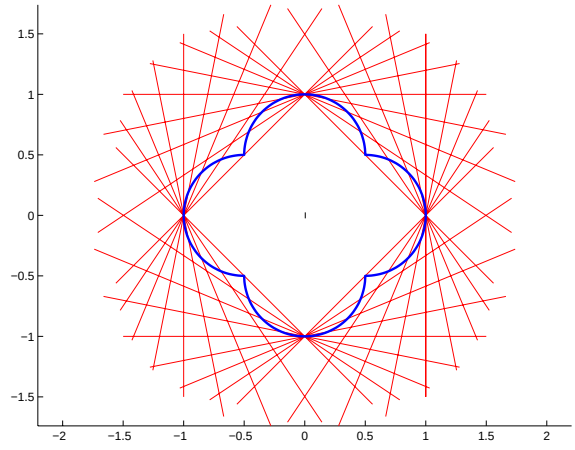


Fig.6.4D

Figure 6.4: Speed profiles: strict convexity, convexity, and smoothness.

- at every point $\boldsymbol{x} \in \Gamma_0$ draw the "flipped" vehicle's speed profile $\rho = g(\boldsymbol{x}, \theta)$ scaled by the factor Δt;
- take the envelope of all these profiles to obtain an approximation of $\Gamma_{t+\Delta t}$;
- the limit of such approximations as $\Delta t \to 0$ provides an equivalent definition of the front's propagation.

Remark 6.1.5. In the canonical definition (Figures 6.5A and 6.5C) a circle was drawn at every point of the front, but the radius of that circle depended upon the orientation of the front at that point; in the modified definition (Figures 6.5B and 6.5D) the profile drawn

at each point might be more complicated, but its shape and size are independent of the orientation of the front.

Remark 6.1.6. The fact that the "flipping" of the vehicle's speed profile is required stems from the fundamental difference between Huygens' principle and Dijkstra's method (based on the optimality criterion): Dijkstra's method looks for the fastest way to reach the front starting from the given point $\boldsymbol{x}$ (and using the speed profile information supplied at the point $\boldsymbol{x}$), whereas Huygens' principle suggests looking for the time it will take to advance the front past the point $\boldsymbol{x}$ (using the speed profile information supplied at the front).

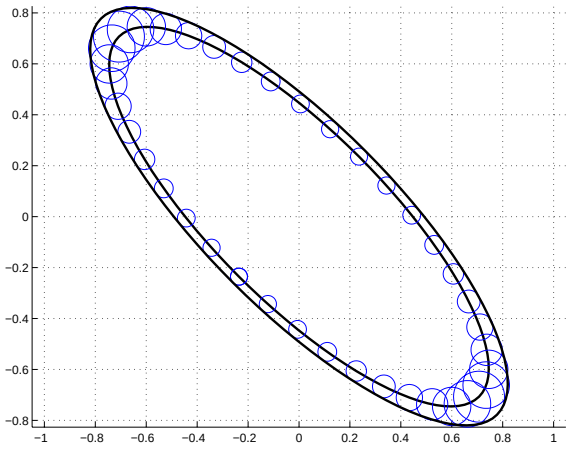


Fig.6.5A: canonical Huygens' construction for an expanding ellipse.

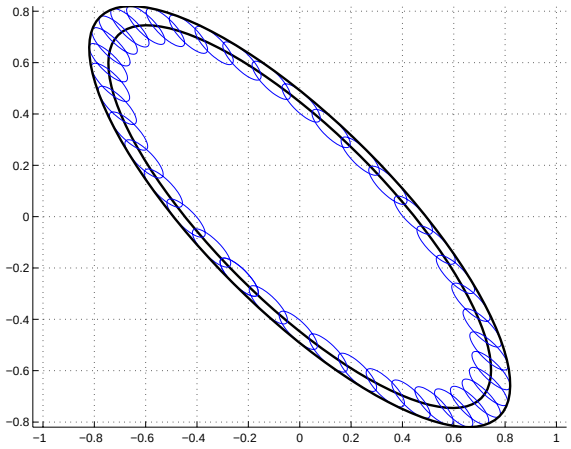


Fig.6.5B: modified Huygens' construction for an expanding ellipse.

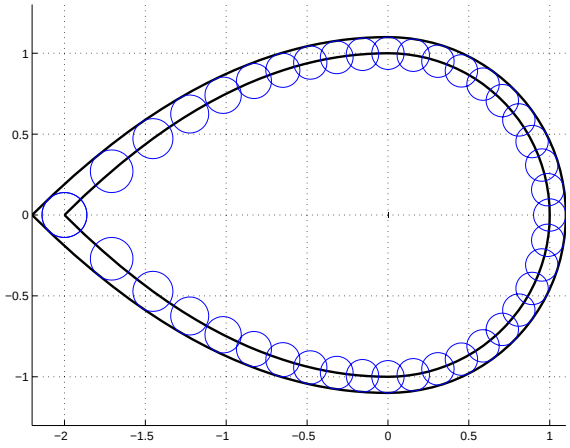


Fig.6.5C: canonical Huygens' construction for an expanding droplet.

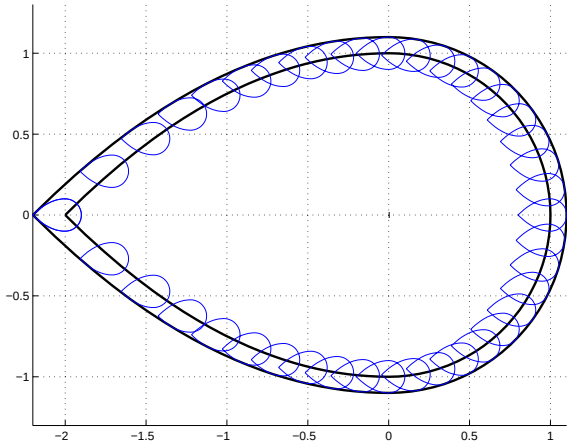


Fig.6.5D: modified Huygens' construction for an expanding droplet.

Figure 6.5: Fronts expanding in homogeneous anisotropic medium: the canonical and modified Huygens' constructions.

6.2 Numerical methods for front propagation problem

The general front propagation problem can be treated numerically using a number of techniques including the marker-particle, volume-of-fluid, and level-set methods. All of them have certain disadvantages, which make it desirable to use a different (more efficient) method for the case when the Hamiltonian is convex and the normal speed F is finite, bounded away from zero, and strictly positive (or strictly negative). As shown above, if F is smooth, the vehicle's speed function f can be defined in terms of F (as in Eqn 6.5), which will yield the equivalent min-time optimal trajectory problem. Therefore, the single-pass control-theoretic Ordered Upwinding Method described in Chapter 4 can be used in this case.

Here, we develop a family of different single-pass methods for this class of front expansion (contraction) problems. The key idea is that any consistent upwind finite difference discretization can be used to compute an update-from-a-single-simplex $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$. Our derivation of such discretizations generalizes the approach used in defining the Fast Marching Method on unstructured meshes (section 3.4.3).

6.2.1 Upwind finite difference discretization

Consider an unstructured triangulated mesh X of diameter h (i.e., if the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent then $\|\boldsymbol{x_j} - \boldsymbol{x_k}\| \leq h$). Let $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ be two adjacent mesh points and choose some other mesh point $\boldsymbol{x} \in \Omega \backslash \partial\Omega$. Define the unit vectors $\boldsymbol{P_1} = \frac{\boldsymbol{x}-\boldsymbol{x_j}}{\|\boldsymbol{x}-\boldsymbol{x_j}\|}$ and $\boldsymbol{P_2} = \frac{\boldsymbol{x}-\boldsymbol{x_k}}{\|\boldsymbol{x}-\boldsymbol{x_k}\|}$. Assume that $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ are linearly independent and consider the 2×2 nonsingular matrix P having $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ as its rows. Let $v_r(\boldsymbol{x})$ be the value of the directional derivative for the direction $\boldsymbol{P_r}$ evaluated at the point $\boldsymbol{x}$. Assuming that the function u is differentiable at $\boldsymbol{x}$, we have $P\nabla u(\boldsymbol{x}) = \boldsymbol{v}(\boldsymbol{x})$, where $\boldsymbol{v}(\boldsymbol{x}) = \begin{bmatrix} v_1(\boldsymbol{x}) \\ v_2(\boldsymbol{x}) \end{bmatrix}$. Recall that the front propagation Eqn 3.2 can be written as $\|\nabla u(\boldsymbol{x})\|^2 F^2 \left(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}\right) = 1$, which can be restated in terms of $\boldsymbol{v}(\boldsymbol{x})$:

$$\boldsymbol{v}(\boldsymbol{x})^T (PP^T)^{-1} \boldsymbol{v}(\boldsymbol{x}) F^2 \left(\boldsymbol{x}, \frac{P^{-1}\boldsymbol{v}(\boldsymbol{x})}{\|P^{-1}\boldsymbol{v}(\boldsymbol{x})\|}\right) = 1. \tag{6.7}$$

To obtain the discretized equation, we now replace each v_r with the corresponding difference approximation: $v_r(\boldsymbol{x}) \approx a_r U + b_r$, where b_r's linearly depend on the values of U (and possibly of ∇U for higher order schemes) at the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$.

Remark 6.2.1. The simplest first order finite difference approximation is obtained by choosing

$$a_1 = 1/\|\boldsymbol{x} - \boldsymbol{x_j}\|; \qquad b_1 = -U(\boldsymbol{x_j})/\|\boldsymbol{x} - \boldsymbol{x_j}\|;$$
$$a_2 = 1/\|\boldsymbol{x} - \boldsymbol{x_k}\|; \qquad b_2 = -U(\boldsymbol{x_k})/\|\boldsymbol{x} - \boldsymbol{x_k}\|.$$

For convenience, let $Q = (PP^T)^{-1}$ and use $\boldsymbol{v}(\boldsymbol{x}) \approx \boldsymbol{w}(\boldsymbol{x}) = V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b}$. Then the discretized version of Eqn 6.7 can be used as an equation for the upwind-update-from-a-single-simplex $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$:

$$\left((\boldsymbol{a}^T Q \boldsymbol{a})\left(V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})\right)^2 + (2\boldsymbol{a}^T Q \boldsymbol{b}) V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) + (\boldsymbol{b}^T Q \boldsymbol{b})\right) F^2\left(\frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|}, \boldsymbol{x}\right) = 1. \quad (6.8)$$

Remark 6.2.2. In the isotropic case, the analogous equation was just a quadratic (see section 3.4.3). The equation 6.8 is a more complex non-linear equation, since $\boldsymbol{w}(\boldsymbol{x})$ also depends on $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$. In general, this equation will have to be solved approximately and the overall efficiency of the method will also depend on the iterative numerical method used to solve equation 6.8. Since these iterations are generally unavoidable, we will consider solving this equation as a single operation in the further analysis of computational complexity.
We note that the iterative zero-finding required to compute $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is **local** (i.e., $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ can be computed independent from any other $V_{\boldsymbol{x_i},\boldsymbol{x_l}}(\boldsymbol{x_m})$) and, thus, should not be confused with the iterations necessary to solve a coupled system of non-linear equations (such as 2.27) simultaneously.

6.2.2 Upwinding criteria

> By indirections find directions out.
>
> WILLIAM(?) SHAKESPEARE(?) (1564(?) - 1616(?))
> *Polonius' motto, "Hamlet," act II., (66).*

We need to ensure that the value of $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ computed from equation 6.8 is truly upwind, i.e., that the characteristic for the mesh point $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x}\boldsymbol{x_j}\boldsymbol{x_k}$.[9] The approximate gradient $P^{-1}\left(V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b}\right)$ can be used to compute an approximation to the characteristic direction $\boldsymbol{a}(\boldsymbol{x})$. Section 6.1 illustrates how this mapping of ∇u to $\boldsymbol{a}$ (or

[9] As shown in section 4.1, this is different from ensuring that the gradient points from that simplex: the gradient lines of the viscosity solution need not coincide with the characteristic lines of the Hamilton-Jacobi PDE for the anisotropic problems.

ω to α) can be performed analytically in R^2; analogous formulas can be devised for higher dimensions.

For R^n, the requirement that the characteristic direction should point into the simplex $\boldsymbol{x}\boldsymbol{x_1}\ldots\boldsymbol{x_n}$ is equivalent to the condition that all the elements of the vector $(P^T)^{-1}\boldsymbol{a}(\boldsymbol{x})$ should be positive.

Remark 6.2.3. The necessity to impose upwinding criteria stems from the fact that the finite differences update procedure (section 6.2.1) is based on approximating the gradient. The control-theoretic update formula 4.7 is based on solving the local optimization problem directly (i.e., an approximate optimal control within a given simplex is found); thus, no additional upwinding criteria is necessary.

6.2.3 Combined upwind update formula

The unfortunate feature of the upwinding criteria described above is that they are based on the approximate rather than the exact characteristic direction. It is possible that, due to the approximation error, an upwinding criterion will not be satisfied even though the true characteristic for the mesh point $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x}\boldsymbol{x_j}\boldsymbol{x_k}$. If that simplex is small enough, this can happen only when one of the elements of the vector $(P^T)^{-1}\boldsymbol{a}(\boldsymbol{x})$ is close to zero, i.e., only when the characteristic direction almost coincides with $(-\boldsymbol{P_1})$ or $(-\boldsymbol{P_2})$. That corresponds to the situation when $U(\boldsymbol{x})$ can be computed based on either $U(\boldsymbol{x_j})$ or $U(\boldsymbol{x_k})$. Thus, we define the "one-sided-update" formula in the manner consistent with the control-theoretic perspective:

$$V_{\boldsymbol{x_i}}(\boldsymbol{x}) = \frac{\|\boldsymbol{x_i} - \boldsymbol{x}\|}{F\left(\boldsymbol{x}, \frac{\boldsymbol{x_i}-\boldsymbol{x}}{\|\boldsymbol{x_i}-\boldsymbol{x}\|}\right)} + U(\boldsymbol{x_i}). \tag{6.9}$$

So, the final formula for the upwind-update-from-a-single-simplex becomes

$$V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) = \begin{cases} \text{solution of Eqn 6.8} & \text{if } \boldsymbol{P_1} \text{ and } \boldsymbol{P_2} \text{ are linearly independent} \\ & \text{and the upwinding criteria are satisfied;} \\ \min\left(V_{\boldsymbol{x_j}}(\boldsymbol{x}),\ V_{\boldsymbol{x_k}}(\boldsymbol{x})\right) & \text{otherwise.} \end{cases} \tag{6.10}$$

Using the finite difference update formula 6.10 instead of the formula 4.7 in the algorithm described in section 4.3, we obtain a new Ordered Upwinding Method for solving the front expansion problem (Hamilton-Jacobi PDE 3.2). In fact, this defines a whole family of such methods, since different upwind finite difference operators can be used to approximate $w_r(\boldsymbol{x})$ in equation 6.7.

We note that the resulting methods

- are "single-pass" and have the same computational complexity as the method introduced in section 4.3;
- work equally well on acute and non-acute triangulated meshes (no additional splitting section construction is required for the non-acute meshes);
- are applicable for a general anisotropic optimal trajectory problem described in section 2.2.
- can be easily extended to R^n and manifolds (the generalizations of the mapping $\boldsymbol{n} \mapsto \boldsymbol{a}$, of equation 6.7 and of the upwinding criterion are obvious).

Remark 6.2.4 (Convergence). As of right now we do not have a proof of convergence for these methods. We rely on general convergence considerations (see the remarks following the proof of theorem 5.2.1) and on the numerical evidence. In all of our numerical experiments the numerical solution U produced by these methods converges to the viscosity solution of the original PDE. The rate of convergence depends on the particular finite difference operators used to approximate $w_r(\boldsymbol{x})$ in equation 6.7.
More information on the efficiency of these methods can be found in Chapter 7.

Chapter 7

Efficiency, Alternatives, Limitations

> Idealism is fine; but as it approaches reality, the cost becomes prohibitive.
>
> WILLIAM F. BUCKLEY, JR.

An efficient implementation of the described numerical methods for the anisotropic optimal-trajectory and front-propagation problems requires dealing with several algorithmic issues. Storing and sorting the current *AcceptedFront*, for example, has to be implemented rather carefully[1] to enable efficient search for the "*AcceptedFront* neighborhood" NF($\boldsymbol{x}$) for every *Considered* point $\boldsymbol{x}$. The inverse operation (searching for all *Considered* $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$ is another major component of the implementation. Efficient use of data structures allows us to construct an algorithm with the computational complexity of $O(\Upsilon M \log(M))$.

7.1 Two update formulae: trajectory approximation vs. upwind finite differences.

The connection between a particular class of anisotropic front propagation and optimal trajectory problems allows us to build both control-theoretic and hybrid (finite differences based) single-pass methods. On a fixed mesh X, the computational complexity of these

[1] This is particularly true for the implementation of the method in higher dimensions.

methods will be the same. However, the overall efficiency of each program will be affected by the chosen upwind-update-from-a-single-simplex formula. The optimal choice depends on the particular speed functions F and f and on the details of implementation.

Recall that the control-theoretic approach requires performing a local minimization at each mesh point (Eqn 4.7), whereas the finite differences upwind update formula requires finding the roots of the non-linear Eqn 6.8. Sometimes one of these (but not the other) can be performed analytically and, thus, leads to a more efficient program. Generally, however, both the minimization and the root-finding have to be done approximately and the overall efficiency depends on the particular numerical method used to compute the approximate update.

It should also be noted that the above complexity and efficiency discussion is limited to finding a numerical solution on a fixed grid. The speed of convergence (of the numerical approximation $U(\boldsymbol{x})$ to the viscosity solution $u(\boldsymbol{x})$ as the grid is refined) is a separate issue. Thus, the availability of the higher order accurate upwind update formulas is a significant advantage of the hybrid approach.

7.2 Heuristic techniques

> The first time, it's a KLUDGE!
> The second, a trick.
> Later, it's a well-established technique!
>
> *A fundamental principle of software engineering.*

7.2.1 Heuristic Techniques: "Update Relaxation"

In the algorithm described in section 4.3, there are two different situations when the tentative value $V(\boldsymbol{x})$ is recomputed for a *Considered* point $\boldsymbol{x}$:

- $V(\boldsymbol{x})$ is first computed using the entire $\mathrm{NF}(\boldsymbol{x})$ at the moment when $\boldsymbol{x}$ is added to *Considered*;
- $V(\boldsymbol{x})$ is then recalculated from at most 2 simplexes every time the newly *Accepted* mesh point $\bar{\boldsymbol{x}}$ belongs to $\mathrm{NF}(\boldsymbol{x})$.

If the boundary condition for the PDE is nearly constant (i.e., if $q_2 \leq q_1 + hK_2$, where h is the diameter of the triangulated mesh), Lemma 5.1.4 shows that the AF will also approximate the level set throughout the execution of the algorithm. On the other hand, Lemma 2.2.11 shows that the optimal trajectory for $\boldsymbol{x}$ intersects a level set at some point $\tilde{\boldsymbol{x}}$ such that

$$\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq d_1 \Upsilon,$$

where d_1 is the distance from $\boldsymbol{x}$ to that level set. This means that if AF were exactly the level set, the initial evaluation of $V(\boldsymbol{x})$ would capture all the necessary information about all the potential characteristic directions for $\boldsymbol{x}$; thus, the further re-evaluations of $V(\boldsymbol{x})$ would not be necessary. Since AF is only approximating the level set, capturing all the necessary directions requires "widening" the set $\mathrm{NF}(\boldsymbol{x})$. Carefully combining Lemmas 5.1.4 and 2.2.11, we can show[2] that all the characteristic directions are still covered if $\mathrm{NF}(\boldsymbol{x})$ is taken to be two times "wider":

$$\widehat{\mathrm{NF}}(\boldsymbol{x}) \;=\; \{\boldsymbol{x_j x_k} \in AF \;\mid \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_j x_k} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq 2h\Upsilon\}\,.$$

The numerical experiments indicate that a much smaller "widening" (by a single h) is sufficient in practice[3]. This provides for a substantial speed up of the algorithm since no subsequent recomputations of $V(\boldsymbol{x})$ are necessary.

Furthermore, an additional update relaxation can be used with the hybrid methods if the boundary condition for the PDE is nearly constant. In the initial computation of $V(\boldsymbol{x})$ it is often not necessary to consider the entire $\mathrm{NF}(\boldsymbol{x})$. We can stop as soon as we found $\boldsymbol{x_j x_k} \in \mathrm{NF}(\boldsymbol{x})$ such that $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ satisfies the upwinding conditions (see section 6.2.2). The viscosity solution u of Hamilton-Jacobi PDE is Lipschitz-continuous, and, therefore, ∇u exists almost everywhere. As shown in section 6.1, if u is differentiable at the point $\boldsymbol{x}$ and the vehicle's speed profile $\rho = f(\boldsymbol{x}, \theta)$ is strictly convex, then there exists a unique optimizing control $\boldsymbol{a}(\boldsymbol{x})$. Thus, there should not be multiple simplexes in $\mathrm{NF}(\boldsymbol{x})$ producing the updates which satisfy the upwinding criteria[4].

[2] Strictly speaking, Lemma 2.2.11 applies only to the viscosity solution u rather than the numerical solution U. But that is why this is a heuristic technique and not a provably convergent numerical method....

[3] This is not surprising since the upper bounds in Lemmas 5.1.4 and 2.2.11 are derived for the worst case scenario using the global anisotropy coefficient Υ.

[4] The above argument cannot be applied near the shocks. Fortunately, the larger local approximation errors committed near the shock do not affect the stability of the algorithm since those errors are not propagated: the algorithm tracks the characteristic directions, and the characteristics might run into a shock, but never originate from the shock.

For a fixed grid X, the numerical evidence suggests that the "relaxation" significantly improves efficiency of the program. As the grid is refined, however, the numerical solution obtained by the "relaxed" scheme converges to the viscosity solution slower than the numerical solution computed by the "full-update" scheme. Nevertheless, **the rate of convergence** of the "relaxed" and "full-update" schemes seems to be the same.

7.2.2 Heuristic Techniques: "Lifting-to-Manifold"

As described in section 3.5, some non-Eikonal Hamilton-Jacobi equations in the plane can be restated as Eikonal equations on certain manifolds. This statement can be revisited now from the control-theoretic perspective.

Consider an Eikonal equation on a smooth manifold $z = g(x, y)$. The Eikonal equation corresponds to an isotropic speed function f and, therefore, at every point on the manifold the vehicle's speed profile is a circle. Since the Eikonal equation is considered in the intrinsic manifold coordinates, the vehicle's speed profile "lives" in the manifold's tangent plane; thus, its projection onto the $z = 0$ plane is an ellipse. In Chapter 4 we have shown how the original Eikonal problem on the manifold can be replaced by the corresponding anisotropic optimal trajectory problem in the $x - y$ plane.

Discussion in section 3.5 proceeds in the opposite direction: it is observed that some anisotropic problems in the plane can be restated as the equivalent isotropic (Eikonal) problems on certain manifolds. The Fast Marching Method can then be applied to solve the latter (Eikonal) problems on the unstructured meshes approximating the corresponding manifolds. The control-theoretic reasoning reveals a necessary condition for this substitution to be possible: the vehicle's speed profile in the original anisotropic problem has to be an ellipse at every point. This is not a sufficient condition; there is also an issue of the "equality of mixed partial derivatives" of g. It is easy to show an example of optimal trajectory problem where f is smooth and the vehicle's speed profile is an ellipse everywhere, yet there exists no manifold $z = g(x, y)$ corresponding to it.

Since the vehicle's speed profile $\rho = f(\boldsymbol{x}, \theta)$ is an ellipse, the corresponding anisotropy coefficient is

$$\Upsilon = \frac{f_2}{f_1} = \frac{\text{length of the bigger semi-axis of the ellipse}}{\text{length of the smaller semi-axis of the ellipse}}.$$

Thus, the above described "lifting-to-manifold" technique can be interpreted as a reduction of the anisotropy coefficient.

Of course, if this substitution is possible then the general method is not needed since the problem can be efficiently handled by the original Fast Marching Method on the unstructured mesh. However, even if the problem is not reducible to the Eikonal equation on a manifold, it may still be possible to find a function g such that the biggest speed/smallest speed ratio will be much smaller on the manifold $z = g(x, y)$. This can clearly affect the efficiency of the program since the computational complexity of the method is proportional to Υ.

Remark 7.2.1. We note that the Ordered Upwinding Methods described in chapters 4 and 6 can be used on any unstructured mesh. The only part of the program which needs to be modified to use the method on meshes approximating manifolds is the algorithm for sorting and searching the *AcceptedFront*.

Finding the optimal manifold can prove to be as hard as the original problem, but any (heuristically found) $g(x, y)$ which reduces the Υ will improve the performance. A substantial amount of preprocessing can often be justified: for example, in cases when one has to solve numerous optimal trajectory problems with the same running cost function, but with the different boundary conditions.

7.2.3 Heuristic Techniques: using local anisotropy coefficient.

So far we have always used the global bounds on the speed function

$$0 < F_1 \leq F(\boldsymbol{x}, \boldsymbol{p}) \leq F_2,$$

for all $\boldsymbol{p}$ and $\boldsymbol{x}$. We now define the local bounds on F,

$$F_1(\boldsymbol{x}) = \min_{\boldsymbol{p} \in S_1} F(\boldsymbol{x}, \boldsymbol{p}), \qquad F_2(\boldsymbol{x}) = \max_{\boldsymbol{p} \in S_1} F(\boldsymbol{x}, \boldsymbol{p}),$$

and the local anisotropy coefficient $\Upsilon(\boldsymbol{x}) = \frac{F_2(\boldsymbol{x})}{F_1(\boldsymbol{x})}$.

We note that many of the Lemmas stated in Chapter 2 for the value function $u(\boldsymbol{x})$ in terms of F_1 and F_2 can be restated in terms of $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$. Most importantly, this is true for the Remark 2.2.23, which establishes a bound on the angle between the characteristic and gradient directions. Thus, it is also possible to build the numerical method using $\Upsilon(\boldsymbol{x})$ instead of Υ in the definition of $\mathrm{NF}(\boldsymbol{x})$. We also note that if F is smooth and the maximum/minimum in defining $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$ are taken not just at the

point but over some closed ball B centered at $\boldsymbol{x}$, then the resulting algorithm provably converges to the viscosity solution. (Indeed, for small enough h, $\mathrm{NF}(\boldsymbol{x}) \subset B$ even if $NF(\boldsymbol{x})$ were defined using the global anisotropy coefficient Υ.)

This observation leads to a substantially more efficient algorithm because, for the front propagating in a strongly inhomogeneous medium, the global anisotropy coefficient Υ can be much larger than $\sup_{\boldsymbol{x}\in\Omega} \Upsilon(\boldsymbol{x})$.

7.3 Alternatives

Here we discuss two other approaches to the front propagation problems and compare them to the single-pass methods introduced in previous chapters.

7.3.1 Causality for the grid-orientation dependent methods.

We now revisit the simple test problems considered in section 4.1. Recall that Tsitsiklis' algorithm formulated for the isotropic problems is used there to compute the simplest anisotropic front propagation. Our analysis in section 4.1 explains the reasons for the algorithm's failure to converge to the viscosity solution (Figure 4.1B). However, for some anisotropic problems, a numerical solution obtained by an isotropic single-pass methods will actually converge to the correct viscosity solution of the original Hamilton-Jacobi-Bellman PDE.

The reason for the algorithm's success in the computation shown in Figure 4.1A, is the fact that, for this particular orientation of the ellipse, the characteristics happen to lie in the same quadrant as the gradient lines. A criterion based on this observation was introduced by Sethian in [46]:

Criterion 7.3.1 (Applicability of the Fast Marching Method). For a general static Hamilton-Jacobi equation $H(\nabla u, \boldsymbol{x}) = 0$, if the Hamiltonian H is approximated on a Cartesian grid by a consistent difference operator

$$H_{ij}(U_{i,j}, U_{i-1,j}, U_{i+1,j}, U_{i,j-1}, U_{i,j+1}, \boldsymbol{x_{i,j}}) = 0,$$

and if it is known that $U_{i,j}$ depends only on the **smaller** values of U at the neighboring points, then the Fast Marching Method can be used to compute $U_{i,j}$'s efficiently.

Remark 7.3.2. In the context of upwinding discretizations, the above criterion is equivalent to requiring that the characteristics and the (numerically approximated) vector $(-\nabla u)$ should lie in the same quadrant. Several sufficient conditions for a class of numerical Hamiltonians to satisfy the above criterion were presented in [34]. For instance, the causality property was proven in [34] for the Godunov-type upwinding discretization H_{ij}^G, provided the original Hamiltonian $H(\nabla u, \boldsymbol{x})$ has a special form $H(\nabla u, \boldsymbol{x}) = G(u_x^2, u_y^2)$, for some function G. We note that, even for a relatively simple elliptical front propagation equation 4.3, this condition is satisfied only in the case when c_1 or c_2 is equal to zero, i.e., only when the axis of the ellipse are exactly aligned with the grid coordinate directions. This is precisely the situation illustrated by Figure 4.1A.

Originally, a significant part of our research was directed toward finding the discretizations satisfying the criterion 7.3.1. We have discovered, however, that this approach is often impractical due to the following reasons:

- *Whether or not the criterion 7.3.1 is satisfied depends upon a particular grid orientation.* Indeed, the two test problems in Figure 4.1 are actually the same (modulo a rotation by 45^o), yet only one of them satisfies the criterion.

- *For any anisotropic problem, there are infinitely many grid orientations such that the criterion is not satisfied.* If an angle between the characteristic and the gradient line is not zero, then any grid line lying inside that angle will violate the criterion. Correspondingly, the bigger the *anisotropy coefficient* Υ is, the harder it is to find the grid orientation satisfying the criterion.

- *The criterion is infinitely sensitive to grid perturbations.*

- *If the criterion is not satisfied, the numerical solution does not lose stability under grid refinement.* In other words, when it does not work, it is not immediately obvious.

- *If the criterion is not satisfied even at a single grid point, the numerical solution need not converge to the viscosity solution.* The criterion 7.3.1 is the basis for determining the order for computing the values of U. Computing even one of them from a wrong quadrant can greatly affect the ordering of the remaining computations.

- *For many anisotropic problems, the criterion cannot be satisfied for any choice of the grid directions.* Indeed, if the angle between gradient lines and the characteristics is

sufficiently wide, and if the medium is substantially inhomogeneous (i.e., if the speed $f(\boldsymbol{x}, \boldsymbol{a})$ varies significantly in different parts of Ω), then any Cartesian grid might violate the criterion 7.3.1 for some grid point $\boldsymbol{x} \in X$.

As a result, we have chosen to concentrate on a family of robust single-pass methods, which are independent of the grid choice[5] and applicable to a wider class of control problems.

7.3.2 Level Set Methods

Level Set Methods, introduced in [35], rely in part on the theory of curve and surface evolution given in [40, 41] and on the link between front propagation and hyperbolic conservation laws discussed in [42]. They recast interface motion as a time-dependent Eulerian initial value partial differential equation, and rely on viscosity solutions to the appropriate differential equations to update the position of the front, using an interface velocity that is derived from the relevant physics both on and off the interface.

In order to track the evolution of some curve Γ_t (as defined in section 3.1), the Level Set Method uses a function $\phi : R^2 \times R_+ \mapsto R$ such that $\phi(\boldsymbol{x}, 0)$ is the signed distance from $\boldsymbol{x}$ to the curve Γ_0 and

$$\phi_t + F\|\nabla\phi\| = 1. \tag{7.1}$$

This level set equation was first presented by Osher and Sethian in [35]. If ϕ is the viscosity solution of equation 7.1, then the zero level set of ϕ corresponds to the position of the curve Γ_t at all times:

$$\phi(\boldsymbol{x}, t) = 0 \quad \Leftrightarrow \quad \boldsymbol{x} \in \Gamma_t.$$

The equation 7.1 is discretized using upwinding finite difference operators. The resulting discretization is consistent and monotone. Such schemes were proven to converge to the viscosity solution by Crandall and Lions in [16].

This "initial value" formulation (leading to the time dependent Hamilton-Jacobi equation 7.1) has an important advantage when compared to the "boundary value" formulation (leading to the static Hamilton-Jacobi equation 3.2). It can be used on a much wider

[5] Of course, it is just the fact of *convergence* that is independent of the grid choice for our methods; the *speed of convergence* is certainly influenced by the choice of the grid and its alignment with the shock lines; see the numerical experiments in section 8.1.1, for example.
In fact, if the computational mesh is not fixed due to some application-specific reasons, the convergence of our single-pass methods can be further improved by using the computed characteristic information to dynamically add the mesh points inside the AF, wherever the shock is suspected.

class of problems: the function F can change sign and the Hamiltonian $H(\nabla\phi, \boldsymbol{x}) = F\|\nabla\phi\|$ does not have to be convex.

Remark 7.3.3. In [46] it is recommended that all problems, where F depends on the orientation of the front should be framed as "initial value" problems. The reason being, that the classic Fast Marching Method (proposed as a tool for solving the "boundary value" problems) could not be used in anisotropic cases. The general single-pass methods described in the preceding chapters do not have that restriction. Therefore, for the first order Hamilton-Jacobi equations, the "boundary value" formulation can be used efficiently, provided that F is smooth, bounded away from zero, and the Hamiltonian H is convex.

A naive implementation of the Level Set Method (as described in [35]) will be very inefficient for this problem: all of the grid points will have to be updated at each time step. A clever observation (that it is only necessary to track the evolution of a few level sets close to zero) enabled Chopp to formulate the "Narrow Band" version of the Level Set Method [13], which was extensively analyzed by Adalsteinsson and Sethian [1], and used in numerous applications [46].

Even though this method is much more efficient it is still a subject to the CFL condition, namely if $0 < F_1 \leq F \leq F_2$ then $\Delta t F_2 \leq \Delta x$. Suppose the "Narrow Band" version of the Level Set Method is used on an $m \times m$ uniform Cartesian grid, and the narrow band is k grid points wide. The computational complexity of one time-iteration in the narrow band is $O(mk)$. At the same time, F_1 controls the upper bound on the number of time-iterations after which the zero level set will pass through every grid point and will leave the computational domain. Combining this bound with the CFL condition and adding the cost of narrow band re-initializations, we obtain the overall complexity of this algorithm as $O(m^2 \Upsilon \log(mk))$.

This complexity is asymptotically comparable with the complexity of Ordered Upwinding Methods for the boundary value formulation of the front propagation problem. However, the overall efficiency of those methods is still much better due to the following considerations:

- The initial value formulation requires forward-integrating in time the function ϕ. The approximation error associated with this numerical integration affects the quality of the numerical solution for the front propagation problem on a particular grid. Thus, it also affects the speed of convergence of the numerical solution under grid refinement.

- As described in section 7.2.3, for the front propagating in a strongly inhomogeneous medium, the use of **local** anisotropy coefficient $\Upsilon(\boldsymbol{x})$ significantly reduces the complexity of the single-pass methods. However, the complexity of the Level Set Method is still controlled by the **global** anisotropy coefficient Υ, which can be much larger than the $\sup_{\boldsymbol{x}\in\Omega}\Upsilon(\boldsymbol{x})$.

Finally, we note that the above discussion of the efficiency was based on comparing **the asymptotic order** of the methods' computational complexity. In practice, the Ordered Upwinding Methods are usually much faster than the Level Set Methods. A detailed description of various extensions of the Level Set Methods as well as a discussion of applicability to more general front propagation problems can be found in [46].

> The only way of finding the limits of the possible is by going beyond them into the impossible.
>
> ARTHUR C. CLARKE

7.4 Limitations (a.k.a. Future Research)

We conclude this chapter by listing the applicability limitations of our single-pass methods described in chapters 4 and 6.

- The Hamiltonian has to be convex.
- The speed functions for the anisotropic front propagation and optimal trajectory problems have to be bounded away from zero.
- Our current proof of convergence requires the continuity of the speed function.
- The rate of convergence of the methods under mesh refinement is not proven.
- The complexity of the methods in R^n is proportional to $\Upsilon^{(n-1)}$.
- Our current implementation of the single-pass methods is inherently sequential.
- The methods have so far been used only for the **deterministic** optimal trajectory problems.

- Only the first-order PDEs can be currently treated, since the characteristic directions and the finite speed of information propagation are essential preconditions for decoupling a system of discretized equations.

We already know that some of these limitations can be relaxed in practice using the heuristic techniques described in section 7.2. We are currently exploring the applicability of Ordered Upwinding Methods to other classes of static partial differential equations.

Chapter 8

Numerical Experiments

In this chapter we will use single-pass methods to solve the front propagation and optimal trajectory problems. The numerical experiments in section 8.1 will illustrate the use of several versions of the Fast Marching Method to solve isotropic problems on triangulated meshes in R^2 and on manifolds. In section 8.2 we will use the Ordered Upwinding Methods introduced in chapters 4 and 6 to treat several non-Eikonal test problems from optimal control, computational geometry, and seismology. While these examples are not as complex as the real applications, each one of them illustrates the role that anisotropy plays in the corresponding application domain.

8.1 Numerical tests: the Eikonal equation.

The practical importance of the single-pass methods for the Eikonal equation stems from the multitude of isotropic processes fully described by this equation in a variety of application domains. Tsitsiklis' algorithm and the related label-correcting methods were used in many control-theoretic problems [7, 50, 10, 11]. Early applications of Sethian's Fast Marching Method included photolithography [44], a comparison with volume-of-fluid techniques [25], and a fast algorithm for image segmentation [32]. Some of the more recent applications of the Fast Marching Method include the problems in robotic navigation [28], extension velocity computation [2], visibility evaluation [45], geophysics [39, 47], and computational geometry [29].

Here we consider only two isotropic problems[1] illustrating the particular extensions

[1] These two experiments were taken from [48].

of the Fast Marching Method described in Chapter 3.

8.1.1 Higher order Fast Marching Method.

We compare the performance of the first order accurate and the higher order accurate implementations of the Fast Marching Method on a triangulated mesh. Both methods are used on a uniform mesh to solve the Eikonal equation $\|\nabla u(\boldsymbol{x})\| = 1$ in R^2. The viscosity solution of this equation taken with zero boundary condition is the "distance from the boundary" function. Two examples with shocks are considered: one where the shock line occurs along the grid lines, and another where the shock line is not aligned with the grid. Both the L_2 and L_∞ errors are computed on the grid. Note that the rate of convergence in the L_2 norm generally corresponds to the order of the difference approximations. However, the rate of convergence in the L_∞ norm might be lower (depending on the location of shocks relative to the grid). This is due to the fact that the higher order approximations are meaningful only where the solution is sufficiently smooth. Fortunately, viscosity solutions are differentiable almost everywhere and no information emanates from the shocks. The upwinding difference approximations ensure that the numerical solution $U(\boldsymbol{x})$ mimics this useful property of the viscosity solution; thus, the L_2 norm convergence is not affected by the larger errors committed near the shocks.

The tests are performed on a grid of equilateral triangles spanning the parallelogram with the vertices at $(0,0)$, $(1,0)$, $(.5,\sqrt{3}/2)$, and $(1.5,\sqrt{3}/2)$. The exact values for the distance are used in the narrow band of radius .1 around the initial points to start the algorithm.

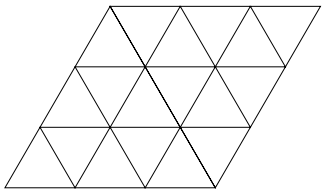
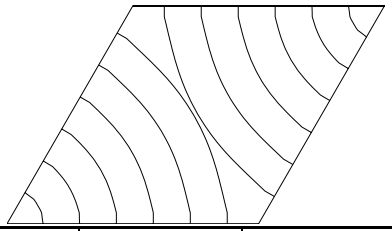


Figure 8.1: The uniform grid of equilateral triangles on a parallelogram.

We first use the Fast Marching Method to compute the distance from two vertices: $(0,0)$ and $(1.5,\sqrt{3}/2)$. The shock line runs along the edges of simplexes (along the shorter diagonal of the parallelogram). The table in Figure 8.2 shows the errors under the mesh

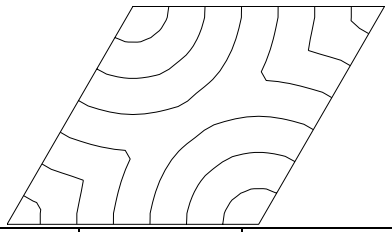
refinement.



Grid	L_2 Error 1st Order	L_2 Error 2nd Order	L_∞ Error 1st Order	L_∞ Error 2nd Order
21^2	0.00818	0.00753	0.01500	0.02161
41^2	0.00432	0.00142	0.00785	0.00559
81^2	0.00213	0.00024	0.00386	0.00130
161^2	0.00108	0.00004	0.00195	0.00031

Figure 8.2: First and "second order" computations of distance from two points. The shock line runs along the edges of simplexes.

We now compute the minimal distance from the other two vertices: $(1, 0)$ and $(0.5, \sqrt{3}/2)$. In this case the shock line is not aligned with the grid lines (edges of simplexes) and it runs along the longer diagonal. The table in Figure 8.3 shows the errors under mesh refinement. The L_2 error is still second-order convergent for the second order scheme, while its L_∞ error is lower order though still much better than the L_∞ error of the first-order scheme. This is to be expected: due to the grid alignment, the approximation is sometimes performed across the shock lines which leads to the first order errors there.



Grid	L_2 Error 1st Order	L_2 Error 2nd Order	L_∞ Error 1st Order	L_∞ Error 2nd Order
21^2	.00645	.00533	.01221	.02034
41^2	.00339	.00117	.00642	.00644
81^2	.00165	.00022	.00315	.00203
161^2	.00084	.00005	.00159	.00071

Figure 8.3: First and "second order" computations of distance from two points. The shock line is not aligned with the edges of simplexes.

8.1.2 Computational geometry: the equidistant offsets.

In manufacturing processes it is often necessary to find the equidistant offsets on a machine part. In the following example we use a triangulated mesh to approximate a complex machine part and then apply the Fast Marching Method to compute the offsets equidistant from the bounding box on that mesh (see Figure 8.4). This computation illustrates the splitting section constructions; the triangulation is obtained by mapping a regular triangular mesh in the $x - y$ plane onto the surface, creating a large number of obtuse and near-degenerate triangles, including some with angles bigger than 160^o.

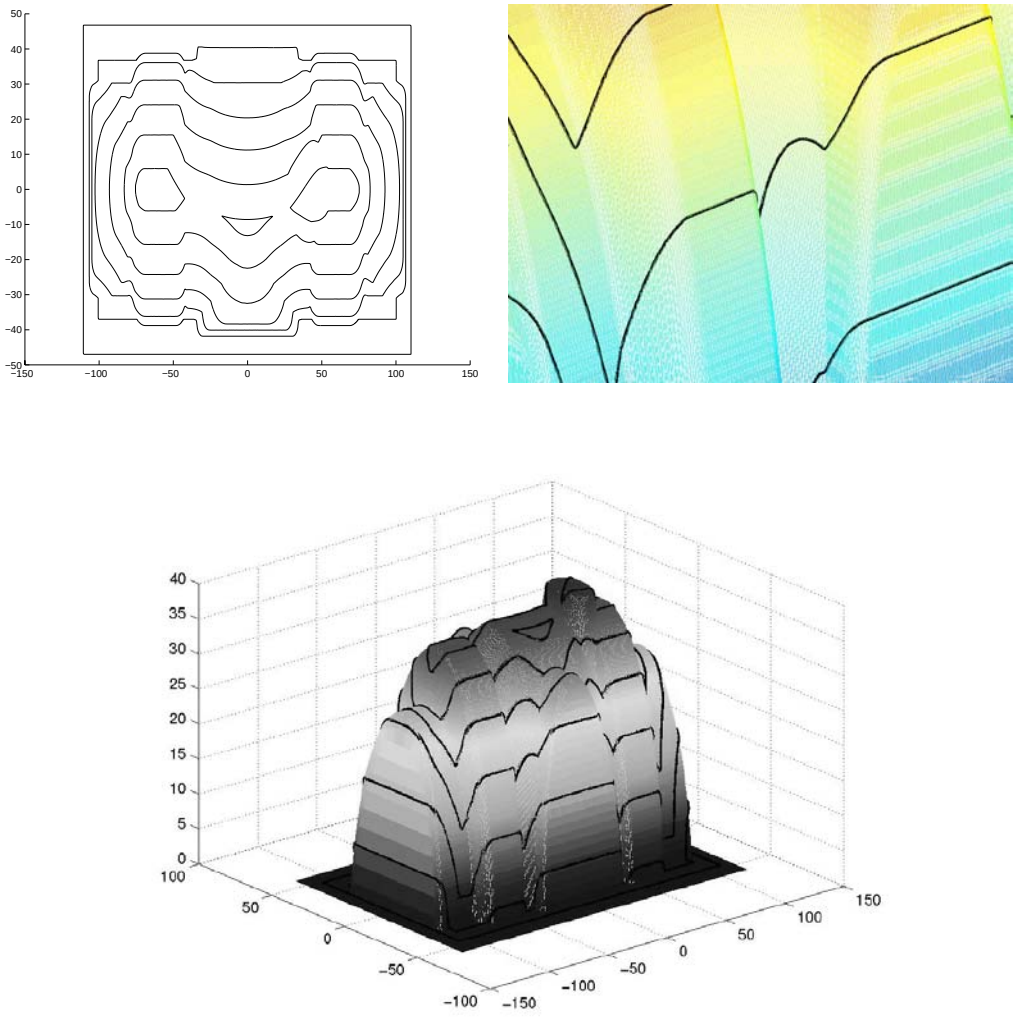


Figure 8.4: Fast Marching Method on a non-acute triangulated mesh.

8.2 Numerical tests: the anisotropic problems.

8.2.1 Geodesic distances on manifolds.

The first test problem is to find the geodesic distance on a manifold $z = g(x, y)$. As described in [27] and [46], this can be accomplished by approximating the manifold with a triangulated mesh and then solving the distance equation $\|\nabla u\| = 1$ on that mesh. Since the latter equation is Eikonal, the Fast Marching Method can be used to solve it efficiently. However, if one desires to formulate the problem in the $x - y$ plane instead of the intrinsic manifold coordinates, then the corresponding equation for u is not Eikonal. Indeed, in the $x - y$ plane, the manifold's geodesic distance function u has to satisfy the equation 3.2 with the speed function F defined as:

$$F(\omega, x, y) = \sqrt{\frac{1 + g_y^2 \cos^2(\omega) + g_x^2 \sin^2(\omega) - g_x g_y \sin(2\omega)}{1 + g_x^2 + g_y^2}}, \tag{8.1}$$

where ω is the angle between $\nabla u(x, y)$ and the positive direction of the x-axis. The degree of anisotropy in this equation is substantial, since the dependence of F upon ω can be pronounced when ∇g is relatively large[2].

As shown in section 6.1, u can also be considered as a value function for the corresponding min-time optimal trajectory problem and must, therefore, satisfy the equation 2.24. The vehicle's speed function $f(\boldsymbol{a}, x, y)$ can be defined applying the formula 6.5 to the speed of front propagation $F(\omega, x, y)$. However, it is even easier to obtain f from the control-theoretic considerations. If the vehicle moving with the speed $f(\boldsymbol{a}, x, y)$ in the $x - y$ plane is just a shadow of another vehicle moving with a unit speed on the manifold, then this vehicle's speed profile is just an orthogonal projection of a unit circle from the manifold's tangent plane onto the $x - y$ plane, i.e.,

$$f(\boldsymbol{a}, x, y) = \left(1 + (\nabla g(x, y) \cdot \boldsymbol{a})^2\right)^{-\frac{1}{2}}, \tag{8.2}$$

where $\boldsymbol{a}$ is a vector of unit length and f is the control-theoretic speed of motion in the direction $\boldsymbol{a}$ (see sections 2.2.5 and 6.1 for details).

[2] The algorithm presented in [27] using the manifold-approximating mesh is more efficient for this problem; here, it serves as a convenient test problem for the general anisotropic case: the numerical solution obtained by the Fast Marching Method on the manifold is compared to the solution obtained by the "general" Ordered Upwinding Method in the $x - y$ plane. We note, of course, that only specific anisotropic problems can be converted into Eikonal equations on manifolds; see [48] for details.

As an example, we consider the manifold $g(x,y) = .75\sin(3\pi x)\sin(3\pi y)$ and compute the geodesic distance on it from the origin (see Figure 8.5). The *anisotropy coefficient* for this problem is $\Upsilon = \frac{F_2}{F_1} = \sqrt{32+81\pi^2}/(4\sqrt{2}) \approx 5.1$. The computations are performed twice:

- using the control-theoretic Ordered Upwinding Method (Chapter 4);
- using the hybrid Ordered Upwinding Method (Chapter 6) with the first order difference approximations.

Refining the mesh and comparing the numerical results to the values of U computed directly on the manifold, we observe the first-order of convergence for both general single-pass methods. In Figure 8.5 we show the level sets of the numerical solution U obtained on a regular mesh with 292×292 mesh points.

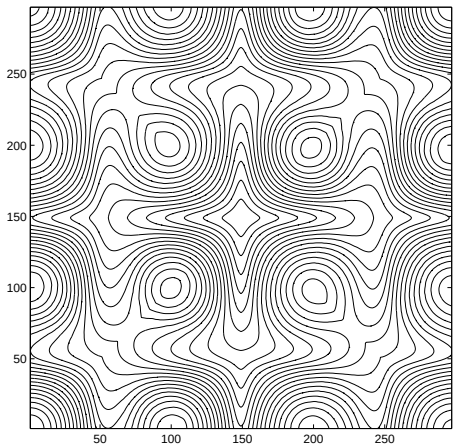


Figure 8.5: The geodesic distance from the origin on the manifold $z = .75\sin(3\pi x)\sin(3\pi y)$ computed on the square $[-.5,.5]\times[-.5,.5]$ in the $x-y$ plane.

8.2.2 Min-time optimal trajectory problem.

Our second example is a particular min-time optimal trajectory problem. Suppose that a vehicle's dynamics $\boldsymbol{z}(t)$ in the square $[-.5,.5]\times[-.5,.5]$ is described by

$$\begin{aligned} \tfrac{d\boldsymbol{z}}{dt}(t) &= \tilde{\boldsymbol{a}}(t) + \boldsymbol{b}(\boldsymbol{z}(t)), \\ \boldsymbol{z}(0) &= \begin{bmatrix} x \\ y \end{bmatrix}, \end{aligned} \tag{8.3}$$

where the advection velocity $\boldsymbol{b} : R^2 \mapsto R^2$ is known and $\tilde{\boldsymbol{a}} : R_{+,0} \mapsto S_1$ is the control. We further assume that $\|\boldsymbol{b}(x,y)\| < 1$ for all $(x,y) \in [-.5,.5] \times [-.5,.5]$. The optimal control $\tilde{\boldsymbol{a}}(\cdot)$ will minimize the time it takes for the vehicle to reach the origin from the point $\boldsymbol{x}$. The value function $u(\boldsymbol{x})$ corresponding to that minimal time is the viscosity solution of the Hamilton-Jacobi-Bellman equation 1.6 with the boundary condition $u(0,0) = 0$. The speed profile for the point $\boldsymbol{x}$ is a unit circle displaced by the vector $\boldsymbol{b}(x,y)$ (see Figure 8.6, for example). The corresponding speed function is

$$f(\boldsymbol{a},x,y) = \boldsymbol{a}\cdot\boldsymbol{b} + \sqrt{(\boldsymbol{a}\cdot\boldsymbol{b})^2 - \boldsymbol{b}\cdot\boldsymbol{b} + 1}, \tag{8.4}$$

where the direction of motion is $\boldsymbol{a} = \frac{\tilde{\boldsymbol{a}}+\boldsymbol{b}}{\|\tilde{\boldsymbol{a}}+\boldsymbol{b}\|}$.

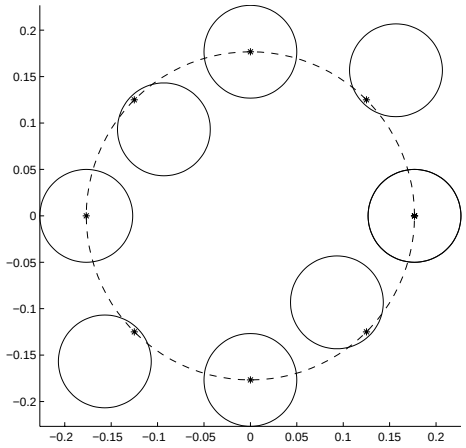


Figure 8.6: The vehicle's speed profiles for 8 different points on the circle.

If $\boldsymbol{b}(x,y) = \begin{bmatrix} b_1(x,y) \\ b_2(x,y) \end{bmatrix}$ then we can rewrite the Hamilton-Jacobi PDE satisfied by $u(x,y)$ as follows:

$$\begin{aligned} (1-b_1^2)u_x^2 + (1-b_2^2)u_y^2 - 2b_1b_2u_xu_y - 2b_1u_x - 2b_2u_y &= 1, \\ u(0,0) &= 0. \end{aligned} \tag{8.5}$$

As an example, we consider a particular advection velocity

$$\boldsymbol{b}(x,y) = \frac{-.9\sin(4\pi x)\sin(4\pi y)}{\sqrt{x^2+y^2}} \begin{bmatrix} x \\ y \end{bmatrix}$$

The "anisotropy coefficient" for this problem is

$$\Upsilon = \frac{F_2}{F_1} = \max_{x,y} \frac{1 + \|\nabla b(x,y)\|}{1 - \|\nabla b(x,y)\|} = 19.$$

The calculations are performed using an Ordered Upwinding Method with the the control-theoretic formula 4.7 for an update-from-a-single-simplex. In Figure 8.7 we show the level sets of the numerical solution U obtained on a regular mesh with 96×96 mesh points.

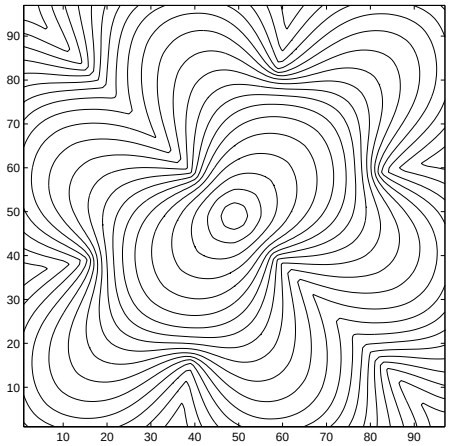


Figure 8.7: The value function for the min-time optimal trajectory problem. The vehicle's speed profile at every point (x, y) is the unit circle displaced by the vector $\boldsymbol{b}(x,y) = \frac{-.9\sin(4\pi x)\sin(4\pi y)}{\sqrt{x^2+y^2}}(x,y)$.

8.2.3 First arrivals in inhomogeneous anisotropic medium.

Finally, we include an example of the first arrival travel times computation with applications to seismic imaging. We start with a computational domain which suggests material layering under a sinusoidal profile. The computational domain is the square $[-a, a] \times [-a, a]$, with layer shapes

$$C(x) = A \sin\left(\frac{m\pi x}{a} + \beta\right) \tag{8.6}$$

where A is the amplitude of the sinusoidal profile, m is the number of periods, and β is the phase offset. The domain is split into n layers by the curves $y_i(x) = X(x) + b_i$, where $i = 1, \ldots, (n-1)$.

In each layer, the anisotropic speed at every point (x, y) is given by an ellipse with bigger axis (of length $2F_2$) tangential to the curve $C(x)$ and the smaller axis (of length $2F_1$)

normal to the curve. F_1 and F_2 are constants in each layer. Thus, the ellipse's orientation and shape depend on (x, y).

This leads to an anisotropic Hamilton-Jacobi equation of the form:

$$\|\nabla u(x,y)\| F = 1, \qquad u(0,0) = 0, \tag{8.7}$$

where the front propagation speed at every point (x, y) is given by the formula

$$F(x,y,u_x,u_y) = F_2 \left(\frac{(1+q^2)u_x^2 + (1+p^2)u_y^2 - 2pqu_xu_y}{(1+p^2+q^2)(u_x^2+u_y^2)} \right)^{1/2}, \tag{8.8}$$

with

$$\begin{bmatrix} p \\ q \end{bmatrix} = \frac{\sqrt{\left(\frac{F_2}{F_1}\right)^2 - 1}}{\sqrt{1 + \left(\frac{dC}{dx}(x)\right)^2}} \begin{bmatrix} \frac{dC}{dx}(x) \\ -1 \end{bmatrix}.$$

Here, F_1 and F_2 are the ellipse semiaxes for the layer corresponding to the point (x, y).

These calculations are performed using the Ordered Upwinding Methods with the the control-theoretic and finite difference formulae for computing an update-from-a-single-simplex. Both methods produce numerical solutions converging to the value function of the corresponding min-time optimal trajectory problem.

The equi-arrival curves shown in Figure 8.8 are obtained on a 193×193 regular mesh using the following parameter values:

$$a = .5, \qquad A = .1225,$$
$$m = 2; \qquad \beta = 0,$$
$$\text{and layer offsets } b_i = (-.25, 0, 0.25).$$

The max/min speed pair (F_2, F_1) for each layer is given in the figures. We note that in one of these examples the global *anisotropy coefficient* $\Upsilon = \frac{3}{.2} = 15$.

Remark 8.2.1. Since the speed function F is discontinuous across the layer boundaries, the standard viscosity solution results for the Hamilton-Jacobi-Bellman equation [16, 15] are not directly applicable. Thus, our proof of convergence in Chapter 5 is not valid in this case either. Nevertheless, the produced numerical solutions seem to converge to the true value function of the corresponding control problem. This is not surprising since our methods are based on approximating Bellman's optimality principle, which is valid for a value function u under much more general assumptions about the speed (or the cost) of motion. See also remark 5.2.3 in Chapter 5.

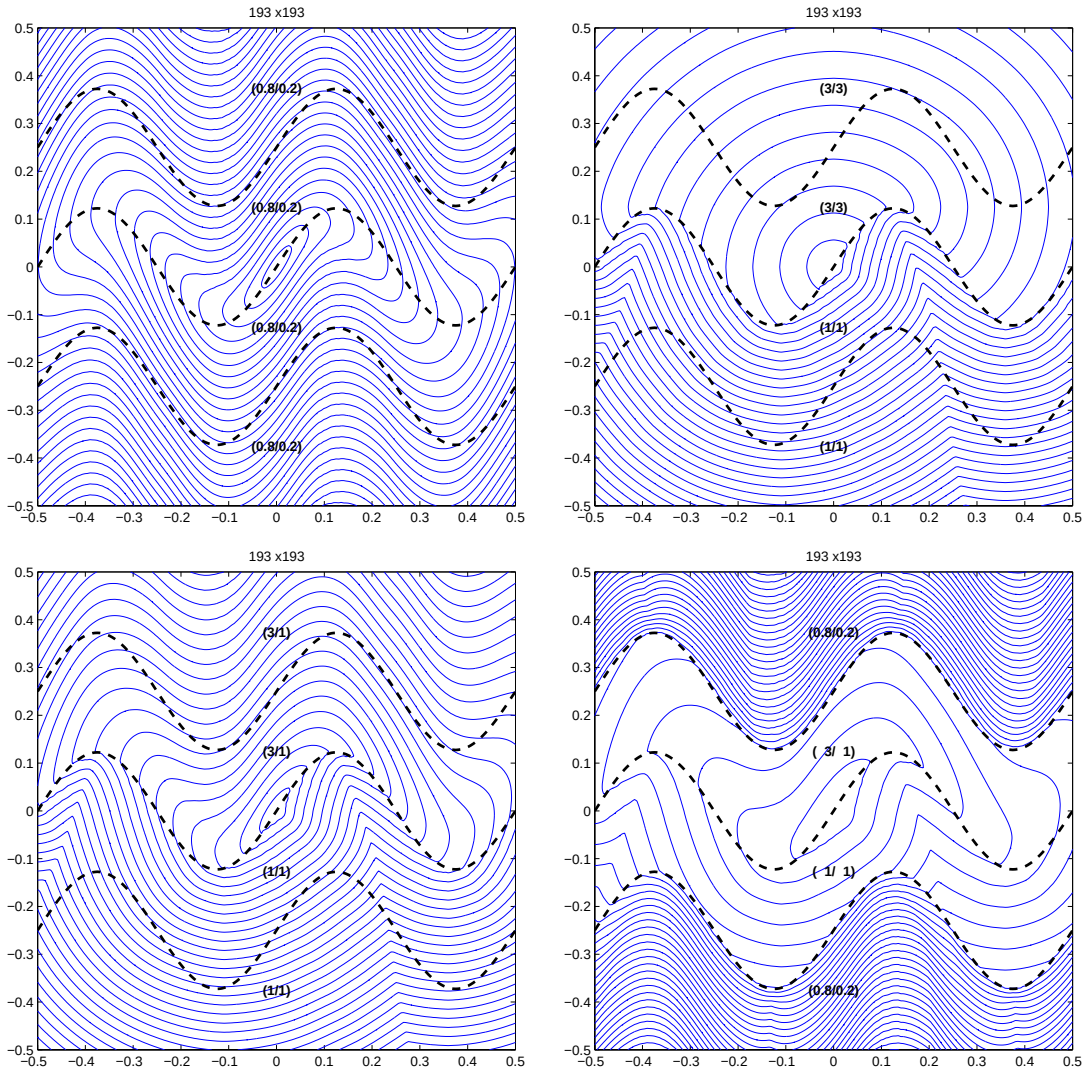


Figure 8.8: Seismic imaging test problem: equi-arrival curves in inhomogeneous, multi-layer medium.

Bibliography

[1] Adalsteinsson, D. & Sethian, J.A., *A Fast Level Set Method for Propagating Interfaces*, J. of Comp. Physics, 118, pp. 269-277, 1995.

[2] Adalsteinsson, D. & Sethian, J.A., *The Fast Construction of Extension Velocities in Level Set Methods*, J. of Comp. Physics, 118, pp. 2-22, 1999.

[3] Ahuja, R.K., Magnanti, T.L., & Orlin, J.B., Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.

[4] Bardi, M. & Falcone, M., *An Approximation Scheme for the Minimum Time Function*, SIAM J. Control Optim., 28, pp. 950-965, 1990.

[5] Bardi, M., Falcone, M., & Soravia, P., Fully Discrete Schemes for the Value Function of Pursuit-Evasion Games, "Advances in dynamic games and applications" (Geneva, 1992), pp. 89–105, Ann. Internat. Soc. Dynam. Games, 1, Birkhuser Boston, Boston, MA, 1994.

[6] Barles, G. & Souganidis, P.E., *Convergence of Approximation Schemes for Fully Nonlinear Second Order Equations*, Asymptotic Analysis, 4, pp. 271-283, 1991.

[7] Beldiman, O., Wang, H.O. & Bushnell, L., *On Trajectory Design for High-Rise High-Speed Elevators*, "Proc. American Control Conference", Philadelphia, PA, p. 3455-3460., 1998.

[8] Bellman, R., Dynamic Programming, Princeton University Press, New Jersey, 1957.

[9] Bellman, R., Introduction to the Mathematical Theory of Control Processes, Academic, New York, 1967.

[10] Bertsekas, D. P., Polymenakos, L. C. & Tsitsiklis, J. N., *Efficient Algorithms for Continuous-Space Shortest Path Problems*, IEEE Transactions on Automatic Control, Vol. 43, pp. 278-283, 1998.

[11] Bertsekas, D. P., Guerriero, F. & Musmanno, R., *Parallel Shortest Path Methods for Globally Optimal Trajectories*, in "High Performance Computing: Technology, Methods, and Applications", (J. Dongarra et.al., Eds.), Elsevier, 1995.

[12] Capuzzo Dolcetta, I. & Falcone, M., *Discrete Dynamic Programming and Viscosity Solutions*, Annales de l'Institut Henri Poincare - Analyse non lineaire, 6, pp. 161-183, 1989.

[13] Chopp, D.L., *Computing Minimal Surfaces via Level Set Curvature Flow*, J. of Comp. Physics, 106, pp. 77-91, 1993.

[14] Capuzzo Dolcetta, I., *On a Discrete Approximation of the Hamilton–Jacobi Equation of Dynamic Programming*, Appl. Math. Optim., 10, pp. 367-377, 1983.

[15] Crandall, M.G., Evans, L.C. & Lions, P-L., *Some Properties of Viscosity Solutions of Hamilton-Jacobi Equations*, Tran. AMS, 282, pp. 487-502, 1984.

[16] Crandall, M.G. & Lions, P-L., *Viscosity Solutions of Hamilton-Jacobi Equations*, Tran. AMS, 277, pp. 1-43, 1983.

[17] Dellinger, J.A., *Anisotropic Seismic Wave Propagation*, Ph.D. Dissertation, Department of Geophysics, Stanford University, CA, 1991.

[18] Dijkstra, E.W., *A Note on Two Problems in Connection with Graphs*, Numerische Mathematik, 1, pp. 269-271, 1959.

[19] Evans, L.C., Partial Differential Equations, American Mathematical Society, 1998.

[20] Evans, L.C. & Souganidis, P.E., *Differential Games and Representation Formulas for Solutions of Hamilton-Jacobi-Isaacs Equations*, Indiana Univ. Math. J., 33, no.5, pp. 773-797, 1984.

[21] Falcone, M., *The Minimum Time Problem and Its Applications to Front Propagation*, in "Motion by Mean Curvature and Related Topics", Proceedings of the International Conference at Trento, 1992, Walter de Gruyter, New York, 1994.

[22] Falcone, M., *A Numerical Approach to the Infinite Horizon Problem of Deterministic Control Theory*, Applied Math. Optim., 15, pp. 1-13, 1987; corrigenda 23, pp. 213-214, 1991.

[23] Falcone, M. & Ferretti, R., *Discrete Time High-Order Schemes for Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, Numerische Mathematik, 67, pp. 315-344, 1994.

[24] Gonzales, R. & Rofman, E., *On Deterministic Control Problems: an Approximate Procedure for the Optimal Cost, I, the Stationary Problem*, SIAM J. Control Optim., 23, 2, pp. 242-266, 1985.

[25] Helmsen, J., Puckett, E.G., Colella, P. & Dorr, M., *Two New Methods for Simulating Photolithography Development in Three Dimensions*, SPIE 1996 International Symposium on Microlithography, SPIE, v. 2726, pp. 253-261, June 1996.

[26] Jain., A., Fundamentals of Digital Image Processing, Prentice-Hall International, NJ, 1989.

[27] Kimmel, R. & Sethian, J.A., *Fast Marching Methods on Triangulated Domains*, Proc. Nat. Acad. Sci., 95, pp. 8341-8435, 1998.

[28] Kimmel, R. & Sethian, J.A., *Fast Marching Methods for Robotic Navigation with Constraints*, Center for Pure and Applied Mathematics Report, University of California, Berkeley, May 1996.

[29] Kimmel, R. & Sethian, J.A., *Fast Voronoi Diagrams and Offsets on Triangulated Surfaces*, AFA Conference on Curves and Surfaces, Saint-Malo, France, July, 1999.

[30] Kruzhkov, S.N., *Generalized Solutions of the Hamilton-Jacobi Equations of the Eikonal Type*, Math. USSR-Sb. 27, pp. 406-445, 1975.

[31] Lesaint, P. & Raviart, P.A., *On a Finite Element Method for Solving the Neutron Transport Equation*, in "Mathematical Aspects of Finite Elements in Partial Differential Equations", (ed. de Boor, C.), Academic Press, New York, 1974.

[32] Malladi, R. & Sethian, J.A., *An $O(NlogN)$ Algorithm for Shape Modeling*, Proc. Nat. Acad. Sci., 93, pp. 9389-9392, 1996.

[33] McGulagh, J., *Geometrical Propositions Applied to the Wave Theory of Light*, Trans. Royal Irish Acad., 17, pp. 241-263, 1837.

[34] Osher, S. & Fedkiw, R.P., *Level Set Methods: An Overview and Some Recent Results*, IPAM GBM Tutorials, March 27 - April 6, 2001.

[35] Osher, S. & Sethian, J.A., *Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton–Jacobi Formulations*, J. of Comp. Physics, 79, pp. 12-49, 1988.

[36] Plimton, S., Hendrickson, B., Shawn, B. & McLendon III, W., *Parallel Algorithms for Radiation Transport on Unstructured Grids*, in proc., 2000.

[37] Postma, G.W., *Wave Propagation in a Stratified Medium*, Geophysics, 20, pp. 780-806, 1955.

[38] Rouy, E. & Tourin, A., *A Viscosity Solutions Approach to Shape-From-Shading*, SIAM J. Num. Anal., 29, 3, pp. 867-884, 1992.

[39] Sethian, J.A., *Fast Marching Methods for Computing Seismic Travel Times*, in progress.

[40] Sethian, J.A., *An Analysis of Flame Propagation*, Ph.D. Dissertation, Department of Mathematics, University of California, Berkeley, CA, 1982.

[41] Sethian, J.A., *Curvature and the Evolution of Fronts*, Commun. in Math. Phys., 101, pp. 487-499, 1985.

[42] Sethian, J.A. (1987) in *Variational Methods for Free Surface Interfaces*, (eds. Concus, P. & Finn, R.), Springer-Verlag, NY, 1987.

[43] Sethian, J.A., *A Fast Marching Level Set Method for Monotonically Advancing Fronts*, Proc. Nat. Acad. Sci., 93, 4, pp. 1591-1595, February 1996.

[44] Sethian, J.A., *Fast Marching Level Set Methods for Three-Dimensional Photolithography Development*, Proceedings, SPIE 1996 International Symposium on Microlithography, SPIE, v. 2726, pp. 262-272, June 1996.

[45] Sethian, J.A., *Fast Marching Methods*, SIAM Review, Vol. 41, No. 2, pp. 199-235, 1999.

[46] Sethian, J.A., Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Sciences, Cambridge University Press, 1996.

[47] Sethian, J.A. & Popovici, M., *Three Dimensional Traveltimes Computation Using the Fast Marching Method*, Geophysics, 64, 2, 1999.

[48] Sethian, J.A. & Vladimirsky, A., *Fast Methods for the Eikonal and Related Hamilton–Jacobi Equations on Unstructured Meshes*, Proc. Nat. Acad. Sci., 97, 11, pp. 5699-5703, 2000.

[49] Schulze, T.P. & Kohn, R.V., *A Geometric Model for Coarsening during Spiral-Mode Growth of Thin Films*, Physica D, 132, pp. 520-542, 1999.

[50] Tronci, E., *On Computing Optimal Controllers for Finite State Systems*, Proc. of 36th IEEE CDC, 1997.

[51] Symes, W. & Qian, J., *Upwind Finite Difference Schemes for Anisotropic Traveltime Computation*, Geophysics, (submitted), 2000.

[52] Taylor, J.E., *Crystalline Variational Problems* Bulletin of the AMS, Vol.84, No.4, pp. 568-588, 1978.

[53] Tsitsiklis, J.N., *Efficient Algorithms for Globally Optimal Trajectories*, IEEE Tran. Automatic Control, 40, pp. 1528-1538, 1995.